Visual Rules Suite - Execution Platform

Execution Server Benutzerhandbuch

Version 5.4.1

Bosch Software Innovations

BOSCH

Americas:
Bosch Software Innovations Corp.
161 N. Clark Street
Suite 3500
Chicago, Illinois 60601/USA
Tel. +1 312 368-2500
info@bosch-si.com
www.bosch-si.com

Asia:
Bosch Software Innovations
c/o Robert Bosch (SEA) Pte Ltd
11 Bishan Street 21
Singapore 573943
Tel. +65 6571 2220
info-sg@bosch-si.com
www.bosch-si.com

Europe:
Bosch Software Innovations GmbH
Ziegelei 7
88090 Immenstaad/GERMANY
Tel. +49 7545 202-300
info-de@bosch-si.com
www.bosch-si.de

Execution Server Benutzerhandbuch: Version 5.4.1

Visual Rules Execution Server 5.4.1
Copyright © 2004-2012 Bosch Software Innovations GmbH

© Bosch Software Innovations GmbH, 2012. Alle Rechte vorbehalten. Weitergabe und Vervielfältigung dieser Publikation oder von Teilen daraus sind, zu welchem Zweck und in welcher Form auch immer, ohne die ausdrückliche schriftliche Genehmigung durch die Bosch Software Innovations GmbH nicht gestattet. MLDS, Visual Rules und Work Frame Relations sind eingetragene Marken der Bosch Software Innovations GmbH. BOSCH und die Bildmarke sind registrierte Marken der Robert Bosch GmbH, Deutschland. Verwendete Produkt- und Firmenbezeichnungen sind eingetragene Marken und - unabhängig von ihrer Kennzeichnung - Eigentum ihrer jeweiligen Inhaber.

Inhaltsverzeichnis

1. Installation und Konfiguration des Execution Server 1

1.1. Distribution 1
1.2. Installation des Execution Server 1
1.3. Aktualisierung einer bestehenden Version 1
1.3.1. Maintenance Tool 1
1.3.2. Migration externer Datenbanken 2
1.3.2.1. Migration externer Datenbanken im interaktiven Modus 2
1.3.2.2. Migration externer Datenbanken mit Kommandozeilen Parameter 2
1.3.3. Migration der eingebetteten Datenbank 3
1.3.3.1. Migration der eingebetteten Datenbank im interaktiven Modus 3
1.3.3.2. Migration der eingebetteten Datenbank mit Kommandozeilen Optionen 3
1.3.4. Migration von der Version 4 3
1.4. Installation der Lizenz 4
1.5. Konfiguration des Execution Server 4
1.5.1. Konfiguration der Authentifizierung und Berechtigungen 5
1.5.1.1. Konfiguration des datei-basierten Security Providers 5
1.5.1.2. JAAS Konfiguration 6
1.5.2. Konfiguration externer Datenbanken 6
1.6. Execution Server Home 7
1.7. Laufzeitprotokollierung 7

2. Erstellung und Bereitstellung von Rule Services 8

2.1. Konzepte 8
2.1.1. Rule Service 8
2.1.2. Regelbibliothek 8
2.1.3. Versionen von Regelbibliothek und Rule Service 9
2.1.4. Visual Rules Archiv 10
2.1.5. Rule Service Einstellungen 10
2.1.5.1. Aktiv 10
2.1.5.2. Gültig von und Gültig bis 10
2.1.5.3. Name der aktiven Konfiguration (Active Configuration Name) 10
2.1.5.4. Statistiklevel 11
2.2. Arbeitsschritte 11
2.2.1. Regeln festlegen, die als Rule Service exportiert werden sollen 11
2.2.2. XML Namespace für Rule Services definieren 12
2.2.3. Einstellen von Aktionen als Rückgabewert 12
2.2.4. Bereitstellen von Regelprojekten vom Visual Rules Modeler 12

3. Arbeiten mit der Webkonsole 15

3.1. Aufruf der Webkonsole 15

3.1.1. Einstellung einer bestimmten Sprache 15
3.2. Verwaltung bereitgestellter Rule Services 16
3.2.1. Anzeige bereitgestellter Rule Services 16
3.2.2. Filterung angezeigter Rule Services 17
3.2.3. Hinzufügen eines Rule Service mittels Visual Rules Archiv 18
3.2.4. Löschen eines bereitgestellten Rule Service 19
3.2.5. Anzeige der WSDL-Datei eines Rule Service 19
3.2.6. Herunterladen eines Rule Service 20
3.2.7. Anzeige der Eigenschaften und Änderung der Einstellungen eines Rule Service 20
3.2.8. Verwaltung der Metadaten eines Rule Service 21
3.2.8.1. Hinzufügen von Metadaten 22
3.2.8.2. Löschen von Metadaten 22
3.2.8.3. Editieren von Metadatennamen und –werten 22
3.2.9. Anzeige der Ausführungen eines Rule Service 23
3.2.10. Anzeige der erforderlichen Bibliotheken eines Rule Service 23
3.3. Verwaltung der Ausführungen von Rule Services 23
3.3.1. Anzeige der Ausführungen von Rule Services 23
3.3.2. Filterung angezeigter Ausführungen 24
3.3.3. Löschen von Ausführungen von Rule Services 25
3.3.4. Herunterladen einer Statistik zur Ausführung eines Rule Service 25
3.4. Verwaltung bereitgestellter Bibliotheken 26
3.4.1. Anzeige bereitgestellter Bibliotheken 26
3.4.2. Filterung angezeigter Bibliotheken 27
3.4.3. Anzeige von Eigenschaften und Verwendung einer Bibliothek 28
3.4.4. Löschen einer bereitgestellten Bibliothek 28
3.5. Wartung des Execution Servers 29
3.5.1. Anzeige von Lizenzinformationen 29
3.5.2. Konfiguration und Anzeige von Nachrichten der Laufzeitprotokollierung 29
3.6. Konfiguration der Anzeige von Tabelleninhalten 30
3.6.1. Ein- und Ausblenden von Spalten 30
3.6.2. Verschieben einer Spalte 30
3.6.3. Verändern des Sortierkriteriums und der Sortierreihenfolge 30
3.7. Filterung angezeigter Objekte 31
3.7.1. Eingabe von Filterkriterien 31
3.7.1.1. Eingabe von Datum und Uhrzeit 31
3.7.2. Anwenden eines Filters 32

4. Aufrufen von Regeln im Execution Server 33

4.1. Konzepte 33
4.1.1. Format der Rule Service Anfrage 33
4.1.2. Format der Rule Service Antwort 34
4.1.3. Generische Rule Service Anfragen 35

4.1.3.1. Generisches VRRequest Format 35
4.1.4. XML Repräsentation von Datentypen 36
4.1.4.1. Einfache Typen 36
4.1.4.2. Strukturen 37
4.1.4.3. Listen und Mengen 37
4.1.4.4. Maps 38
4.1.4.5. Aufzählungen 38
4.1.5. Abbildung des Regelmodells auf WSDL 38
4.2. Arbeitsschritte 39
4.2.1. Aufrufen eines Rule Service 39

5. Rollen des Sicherheitskonzepts 41

5.1. Verfügbare Rollen und Berechtigungen 41

6. Arbeiten mit Metadaten 42

6.1. Konzept Metadaten 42
6.2. Metadaten definieren 42
6.2.1. Zuordnung von Metadaten zu Rule Services 43
6.2.1.1. Standard-Metadata Mapper 43
6.2.1.2. Eine WSDL durch Angabe von Metadata abholen 44
6.2.1.3. Benutzerdefinierter Metadata Mapper 44

Kapitel 1. Installation und Konfiguration des Execution Server

1.1. Distribution

Der Visual Rules Execution Server ist als Web Archive (WAR) gepackt und wird als Bestandteil einer Distribution ausgeliefert. Die Distribution ist ein Zip Archiv und enthält folgendes:

Tabelle 1.1. Distributions Inhalt

Ordner	Inhalt
legalnotice	Rechtliche Informationen zu Lizenzen und Verwendung von Bibliotheken von Drittanbietern
maintenance	Maintenance Tool
manuals	Handbücher

1.2. Installation des Execution Server

Der Execution Server wird als gepacktes Web Archiv (WAR) ausgeliefert und kann installiert werden, indem er auf einem Application Server bereitgestellt wird. Der genaue Vorgang der Bereitstellung hängt vom eingesetzten Application Server ab.

Nach der Bereitstellung auf dem Application Server sollte probehalber die Webkonsole aufgerufen werden. Dies ist in Abschnitt 3.1, „Aufruf der Webkonsole“ beschrieben.

1.3. Aktualisierung einer bestehenden Version

In den folgenden Abschnitten wird davon ausgegangen, dass Java über die Kommandozeile ausführbar ist. Dies kann beispielsweise dadurch erreicht werden, dass die Umgebungsvariable JAVA_HOME gesetzt und auf den Pfad des Betriebssystems genommen wird. Dies ist in der Java Dokumentation beschrieben.

Die Aktualisierung von einer bestehenden Version, wird mit dem Maintenance Tool durchgeführt, welches ein Bestandteil der Execution Server Distribution ist. Die grundsätzlichen Schritte, um dies durchzuführen sind:

1. Backup des verwendeten Datenbank Schemas (dies trifft auch zu, wenn die eingebettete Datenbank verwendet wird) als auch der vorgenommen Konfigurationseinstellungen.
2. Ausführung des Maintenance Tool, um das Datenbank Schema und die Daten zu migrieren
3. Erneutes Bereitstellen der neuen Version des Execution Servers auf dem Application Server
4. Konfiguration vornehmen soweit notwendig

Bestehende Rule Services unterstützen nach der Aktualisierung nicht alle Funktionalitäten der neuen Version. So sind neue Eigenschaften, wie Bereitgestellt Von, hinzugefügt worden, die keine Werte haben.

1.3.1. Maintenance Tool

Das Maintenance Tool ist ein Kommandozeilenprogramm um ein bestehendes Datenbank Schema und die enthaltenen Daten zu migrieren. Es kann ebenso verwendet werden, um Tabellen in einem leeren Datenbank Schema anzulegen. Es ist Bestandteil der Execution Server Distribution und da es als Zip Archiv packetiert ist, muss es als erstes entpackt werden, damit es verwendet werden kann. Zum Aufruf des Programms dient entweder ein *.cmd (für Microsoft Windows Betriebssysteme) oder ein *.sh (für *nix Betriebssystem) Skript. Die Verbindung zur Datenbank verwendet JDBC und benötigt aus diesem Grund den JDBC Treiber für die Datenbank.

Die jar-Datei des Treibers muss dazu nur in das `driver` Verzeichnis kopiert werden, dass sich im Verzeichnis des entpackten Archivs befindet. Dies ist nicht notwendig, wenn die eingebettete Datenbank migriert werden soll.

Nach dem Entpacken kann das Programm auf der Kommandozeile ausgeführt werden. Dazu öffnen Sie eine Kommandozeile und navigieren zu dem Verzeichnis, in dem das Zip Archiv entpackt wurde. Im folgenden Beispiel wird angenommen, dass es sich um das Verzeichnis `maintenance` in `tmp` auf einem Microsoft Windows Betriebssystem handelt:

```
C:\tmp\maintenance>
```

Das Verzeichnis ist richtig, wenn sich das Kommandozeilenprogramm ausführen lässt, beispielsweise durch Eingabe von:

```
MaintenanceTool
```

Dieser Aufruf wird alle verfügbaren Kommandozeilen Optionen mit ihrer Beschreibung und Alternativen ausgeben.

1.3.2. Migration externer Datenbanken

Wenn für den Execution Server bisher eine der unterstützten externen Datenbanken[1] verwendet wurde, dann kann das Maintenance Tool zur Migration verwendet werden.

Vor der Migration sollte zur Sicherheit ein Backup der gegenwärtigen Datenbank erstellt werden. Wie man ein Backup für die verwendete Datenbank erstellt, ist dem entsprechenden Datenbank Handbuch zu entnehmen.

Das Maintenance Tool kann entweder mit allen notwendigen Optionen als Kommandozeilen Parameter ausgeführt werden oder in einem interaktiven Modus, der die Einstellungen abfragt. Der für die Migration verwendete Benutzer der Datenbank, benötigte administrative Rechte, darunter die Berechtigungen zum Anlegen (*create*), Ändern (*alter*) und Löschen (*delete*) von Tabellen.

1.3.2.1. Migration externer Datenbanken im interaktiven Modus

Im interaktiven Modus, werden alle notwendigen Einstellungen abgefragt. Der Modus kann durch nachfolgende Eingabe gestartet werden:

```
MaintenanceTool -i
```

1.3.2.2. Migration externer Datenbanken mit Kommandozeilen Parameter

Es ist möglich das Maintenance Tool als reines Kommandozeilen Programm auszuführen, in dem alle Einstellungen als Parameter auf der Kommandozeile übergeben werden.

Bei der Ausführung mit allen Kommandozeilen Parametern gesetzt, wird das Maintenance Tool die Migration ohne Nachfrage ausführen. Stellen Sie sicher, dass ein Backup davor gemacht wurde.

Ein Schema auf einer MySQL Datenbank könnte beispielsweise mit folgender Kommandozeile migriert werden:

```
MaintenanceTool -d com.mysql.jdbc.Driver -db mysql -user admin -pwd secret
                -url jdbc:mysql://localhost:3306/executionserver
```

Beispiel 1.1. Migration eines Schemas auf MySQL

Die verfügbaren Optionen für die Kommandozeilen Parameter sind:

[1] Für eine Liste der unterstützten Datenbanken lesen Sie bitte in den Execution Server Systemanforderungen nach.

Tabelle 1.2. Kommandozeilen Optionen für die Migration externer Datenbanken

Option	Beschreibung
-d *<driverClassName>*	Voll qualifizierter Klassenname des JDBC Treibers
-db *<dbBrand>*	Fabrikat der Datenbank, die migriert wird
-user *<dbUser>*	Name des Datenbank Benutzers
-pwd *<dbPassword>*	Passwort des Datenbank Benutzers
-url *<jdbcUrl>*	JDBC URL für die Verbindung zur Datenbank
-o *<file>*	Ausgabe der SQL Befehle in eine Datei

Es ist möglich, die SQL Befehle in eine Datei auszugeben anstatt die Migration durchzuführen. Das erzeugte Skript is spezifisch für das Datenbankfabrikat und enthält nur die Befehle, um die analysierte Datenbank zu migrieren.

1.3.3. Migration der eingebetteten Datenbank

Die eingebettete Datenbank, die standardmäßig vom Execution Server verwendet wird, kann ebenfalls mit dem Maintenance Tool migriert werden.

1.3.3.1. Migration der eingebetteten Datenbank im interaktiven Modus

Der interaktive Modus des Maintenance Tools kann benutzt werden um die eingebettete Datenbank zu migrieren. Gestartet wird dieser durch Eingabe von:

```
MaintenanceTool -i -e
```

1.3.3.2. Migration der eingebetteten Datenbank mit Kommandozeilen Optionen

Maintenance Tool kann die eingebettete Datenbank direkt von der Kommandozeile migrieren. Dazu ist die Angabe notwendig, wo sich die eingebettete Datenbank befindet. Das ist ein Ordner mit Namen `vrdbartifacts`, welches sich im Standardfall im Benutzerverzeichnis des angemeldeten Benutzers befindet. Der Ordner kann sich auch in einem anderem Verzeichnis befinden, was durch die Einstellung `visualrules.executionserver.home` bestimmt wird. Das folgende Beispiel zeigt die Verwendung unter der Annahme, dass sich `vrdbartifacts` in `C:\tmp\db` befindet:

```
MaintenanceTool -e -embeddedLocation C:\tmp\db
```

Beispiel 1.2. Migration der eingebetteten Datenbank

1.3.4. Migration von der Version 4

Die Codebasis des Execution Servers hat sich zwischen der Version 4 und 5 geändert. Das betrifft hauptsächlich Anwender, welche einen benutzerdefinierten Metadata Mapper einsetzen. In der nachfolgenden Übersicht sind die wichtigsten Änderungen und Hilfestellungen für die Migration aufgelistet.

- Der Großteil der Änderungen kommen von einer neuen Struktur und Paket Namen. Das kann schnell gelöst werden durch die Anwendung eines Refactorings zum Verwalten der Imports, welches in den meisten modernen Java Entwicklungsumgebungen enthalten ist.
- Die `IMetaDataMapper` Schnittstelle wurde entfernt. Benutzerdefinierte Metadata Mapper Klassen erweitern jetzt die abstrakte Klasse `AbstractMetaDataMapper`. Die Methoden Signaturen haben sich ebenfalls geändert. Anstatt eines Arguments vom Typ `IArtifactStorage` haben diese eins vom Typ `IArtifactStorageReadAccess`. Der `IArtifactStorage` darf in benutzerdefinierten Metadata Mapper Implementierung nicht mehr verwendet werden.
- Veraltete (deprecated) Methoden wurden entfernt.

1.4. Installation der Lizenz

Der Execution Server benötigt eine gültige Lizenzdatei, da ansonsten keinerlei Anfragen bearbeiten werden. Es gibt zwei Möglichkeiten, wie der Server eine Lizenzdatei finden kann:

1. Die Lizenzdatei liegt im Ordner `.visualrules5` im Benutzerverzeichnis des Benutzers der den Server ausführt. Dort wird standardmäßig nach Lizenzdateien gesucht.

2. Es kann ein Dateipfad zur Lizenzdatei mit der Einstellung `visualrules.executionserver.license.file` angegeben werden. Dies wird nur benötigt, wenn sich keine Lizenzdatei im Standardverzeichnis befindet. Lesen Sie Abschnitt 1.5, „Konfiguration des Execution Server“ , um mehr Details darüber zu erhalten, wie dies bewerktstellig werden kann.

Im Fall einer ungültigen oder fehlenden Lizenz wird eine Fehlermeldung auf der Startseite der Webkonsole angezeigt. Ebenso wird eine Fehlermeldung in das Log geschrieben. Dort finden sich ebenfalls Informationen zu den Stellen, an denen der Execution Server nach Lizenzen sucht. Dafür muss Logging eingeschalten und auch für den Level INFO konfiguriert sein.

1.5. Konfiguration des Execution Server

Der Execution Server benötigt keinerlei Einstellungen für die Konfiguration, da er angemessene Standardwerte verwendet. In Produktionsumgebungen ist es ratsam bestimmte Einstellungen, beispielsweise die Datenbank und Punkte in Bezug auf Sicherheit, zu ändern. Um dies durchzuführen, gibt es zwei mögliche Wege:

1. Verwendung der Kontextparameter des Web Containers. Dies sind Name-Wert Paare, welche verwendet werden können, um die bereitgestellte Applikation zu konfigurieren. Dies ist wiederum für jeden Application Server spezifisch. Zum Beispiel erwartet der `Apache Tomcat`, dass diese pro Applikation in der `web.xml` oder global in der `context.xml` Datei definiert werden. Für Details zur Konfiguration der Kontextparameter schauen Sie in der Dokumentation des Host Servers nach.

2. Die Einstellungen können als Eigenschaften in der `visualrules-executionserver.properties` Datei spezifiziert werden, welche im Ordner `config` auf dem Klassenpfad des Execution Servers gefunden werden muss. Dies ist gewöhnlich nicht der bevorzugte Weg.

Bei der Benutzung der Properties Datei müssen unter Umständen bestimmte Zeichen maskiert werden.

Es ist generell nicht ratsam beide Wege gleichzeitig zu verwenden. Sollten jedoch aus irgendeinem Grund beide Konfigurationen aktiv sein, so werden die Kontextparameter gegenüber den Eigenschaften in der `visualrules-executionserver.properties` Datei bevorzugt.

Unten aufgelistet finden Sie die möglichen Optionen zur Konfiguration des Execution Servers.

`visualrules.executionserver.license.file`
: Gültiger Dateipfad zu der Lizenzdatei des Execution Servers. Wird nur benötigt, wenn sich die Lizenz nicht im Standardverzeichnis befindet.

`visualrules.executionserver.localstorage.workingdir`
: Ort, wo der Execution Server Artefakte als Jars speichert. Standardmäßig wird dafür das temporäre Verzeichnis des Web Containers benutzt, welches durch das Kontextattribut `javax.servlet.context.tempdir` bestimmt ist. Der Wert muss ein Pfad zu einem les- und beschreibbaren Ordner sein.

`visualrules.executionserver.home`
: Stellt den Ordner ein, der als Execution Server Home verwendet wird. Der Wert muss ebenfalls ein Pfad zu einem les- und beschreibbaren Ordner sein.

`visualrules.executionserver.security.simple.file`
: Der Pfad einer existierenden, lesbaren Datei, in der Benutzer und Rollen für den datei-basierten Security Provider definiert sind (siehe Abschnitt 1.5.1.1, „Konfiguration des datei-basierten Security Providers“).

`visualrules.executionserver.security.jaas.roledefinition`
: Dieser Wert wird für JAAS (siehe Abschnitt 1.5.1.2, „JAAS Konfiguration“) verwendet, um die Benutzer und ihre Rollen zu beschreiben.

`visualrules.executionserver.security.jaas.loginConfigURL`
Die URL beschreibt die Position der `login.conf` Datei, welche die Definition des `LoginModule` enthält, die von JAAS (see Abschnitt 1.5.1.2, „JAAS Konfiguration“) benutzt werden soll.

`visualrules.executionserver.security.jaas.activated`
Der boolsche Wert kontrolliert ob JAAS (see Abschnitt 1.5.1.2, „JAAS Konfiguration“) aktiviert ist (standardmäßig is der Wert `true`).

`visualrules.executionserver.artifactstorage.jndi.name`
Der JNDI Name zur Benutzung einer externen Datenbank. Siehe Abschnitt 1.5.2, „Konfiguration externer Datenbanken“

`visualrules.executionserver.artifactstorage.db.brand`
Name der verwendeten Datenbank. Siehe Abschnitt 1.5.2, „Konfiguration externer Datenbanken“

`visualrules.executionserver.artifactstorage.db.name`
Veraltet. Wurde zur Angabe der verwendeten Datenbank benutzt und ist durch `visualrules.executionserver.artifactstorage.db.brand` ersetzt.

`visualrules.executionserver.metadata.custom.mapper`
Der voll qualifizierte Klassenname eines benutzerdefinierten Metadata Mappers. Die Benutzung eines benutzerdefinierten Metadata Mappers ist in Abschnitt 6.2.1.3, „Benutzerdefinierter Metadata Mapper“ genauer beschrieben.

`visualrules.executionserver.artifactstorage.transactionManager`
In manchen Umgebungen kann es notwendig sein, den von iBatis verwendeten TransactionManager auszutauschen. Bitte lesen Sie im *Execution Server Integration Handbuch* nach, für welche Umgebungen das Setzen dieser Einstellung erforderlich ist.

`visualrules.executionserver.artifactstorage.transactionManager.defaultAutoCommit`
Diese Einstellung wird an den TransactionManager von iBatis weitergereicht. Bitte lesen Sie im *Execution Server Integration Handbuch* nach, für welche Umgebung das Setzen dieser Einstellung erforderlich ist.

`visualrules.executionserver.artifactstorage.transactionManager.setAutoCommitAllowed`
Diese Einstellung wird an den TransactionManager von iBatis weitergereicht. Bitte lesen Sie im *Execution Server Integration Handbuch* nach, für welche Umgebung das Setzen dieser Einstellung erforderlich ist.

`visualrules.executionserver.artifactstorage.transactionManager.commitRequired`
Diese Einstellung wird an den TransactionManager von iBatis weitergereicht. Bitte lesen Sie im *Execution Server Integration Handbuch* nach, für welche Umgebung das Setzen dieser Einstellung erforderlich ist.

1.5.1. Konfiguration der Authentifizierung und Berechtigungen

Der Execution Server verwendet einen einfachen datei-basierten Security Provider und unterstützt den `Java Authentication and Authorization Service (JAAS)`. Beides ist immer aktiv und wird unterschiedlich konfiguriert.

1.5.1.1. Konfiguration des datei-basierten Security Providers

Der datei-basierte Security Provider wird mit einem Standardbenutzer "admin" mit dem Passwort "admin" ausgeliefert. Dies muss als erstes geändert werden, wenn dieser in einer produktiven Umgebung eingesetzt wird. Um dies durchzuführen, kann der Execution Server so konfiguriert werden, dass die Benutzer aus einer Datei gelesen werden. Siehe Abschnitt 1.5, „Konfiguration des Execution Server“ für Details zur Konfiguration. Die Einträge in der Datei müssen ein bestimmtes Format haben. Der folgende Ausschnitt beschreibt das Format und einen Benutzer.

```
# format: login,password,role1,...,roleN

# known roles:
# - ROLE_ADMIN administrator
# - ROLE_USER any (authenticated) user

#example,pass,ROLE_USER
admin,admin,ROLE_ADMIN,ROLE_USER
```

1.5.1.2. JAAS Konfiguration

Der `Java Authentication and Authorization Service (JAAS)` ist Teil des Java Sicherheitskonzepts und dessen Konfiguration würde den Rahmen dieser Dokumentation sprengen. Eine detaillierte Beschreibung kann in Java SE Security gefunden werden.

JAAS ist extern konfiguriert und wird, wenn es aktiv ist, als erstes nach der Authentifizierung gefragt. Dazu wird der Login Kontextname `ExecutionServerSOA` verwendet. Eine einfache JAAS Konfiguration kann mit der Einstellung `visualrules.executionserver.security.jaas.loginConfigURL` konfiguriert werden, deren Wert eine URL wie beispielsweise `file:/c:/security/SOAServer.config` ist. Dies kann auch mit der Systemeigenschaft `java.security.auth.login.config` erreicht werden. Die Verwendung der Systemeigenschaft wird nicht empfohlen. Die Datei definiert ein `LoginModule`, wie in nachfolgendem Beispiel:

```
ExecutionServerSOA {
   com.example.ExampleLoginModule required debug=false;
};
```

Obwohl die Authentifizierung durch JAAS über das konfigurierte Loginmodul durchgeführt wird, ist es noch notwendig einen Benutzer auf eine Rolle im Server abzubilden. Dies kann dadurch erreicht werden, dass die Einstellung `visualrules.executionserver.security.jaas.roledefinition` konfiguriert wird. Lesen Sie Abschnitt 1.5, „Konfiguration des Execution Server“, um eine Erläuterung zu erhalten, wie der Server konfiguriert werden kann.

Die Abbildung eines Benutzers auf seine Rolle folgt einem spezifischen Format, wo jeder Eintrag mit dem Benutzernamen beginnt, gefolgt von einem Doppelpunkt, gefolgt von den Rollennamen, welche selbst durch ein Komma getrennt sind. Das Ende wird mit einem Semikolon abgeschlossen. Zum Beispiel definiert `admin:ROLE_ADMIN, ROLE_USER; bob:ROLE_USER;` einen Benutzer *admin* mit den Rollen *ROLE_ADMIN* und *RULE_USER* und einen Benutzer *bob* mit der einzigen Rolle *ROLE_USER*.

JAAS kann deaktiviert werden durch setzen der Einstellung `visualrules.executionserver.security.jaas.activated` auf den Wert `false`.

1.5.2. Konfiguration externer Datenbanken

Execution Server verwendet eine eingebettete Apache Derby Datenbank. Für Testzwecke und einfache Szenarien reicht diese zwar, aber unter hoher Last kann diese zu einer schlechten Leistung führen da sie nicht konfiguriert werden kann. Für produktions-ähnliche Systeme lautet die Empfehlung daher, eine externe Datenbank einzusetzen. Das kann durch Verwendung einer Konfiguration per Java Naming and Directory Interface (JNDI) erreicht werden. Der eingesetzte Application Server muss in diesem Fall die eingesetzte Datenbank als Datenquelle unter einem JNDI Namen zur Verfügung stellen. Unter Umständen ist es auch notwendig Änderungen an der bereitgestellten Execution Server `war` Datei vorzunehmen. Das dazu notwendige Vorgehen sollte in der Dokumentation des Application Servers beschrieben sein.

Mit der Einstellung `visualrules.executionserver.artifactstorage.jndi.name` wird der Name der Datenquelle definiert. Zu beachten ist hierbei, dass dies der volle Name sein muss, wie dies in der J2EE Umgebungen eingestellt ist. Ist beispielsweise die Datenbank unter dem Namen `jdbc/mydatabase` verfügbar gemacht worden, so ist im Normalfall der volle Name `java:comp/env/jdbc/mydatabase` einzutragen. Im Zweifel sollte hierzu die Dokumentation des Application Server oder der Administrator des Servers zu Rate gezogen werden.

Der Execution Server unterstützt nur eine bestimmte Anzahl an Datenbanken. Der Name der eingesetzten Datenbank kann mit `visualrules.executionserver.artifactstorage.db.brand` definiert werden. Die möglichen Werte sind: `derby`, `mssql`, `mysql`, `db2` und `oracle`. Lesen Sie Abschnitt 1.5, „Konfiguration des Execution Server“, um eine Erläuterung zu erhalten, wie der Server konfiguriert werden kann.

Mit diesen Einstellungen ist der Execution Server in der Lage sich auf externe Datenbanken zu verbinden und die notwendigen Tabellen für den Betrieb zu erstellen. Dafür ist es erforderlich, dass der verwendete Datenbankbenutzer die Berechtigungen zur Erstellung von Tabellen hat.

Der Execution Server sollte in jedem Fall ein eigenes Datenbank Schema zur Verfügung haben, das weder Tabellen noch Daten enthält. Beim Hochfahren werden Tabellen erzeugt und solche gelöscht, die den gleichen Namen haben wie eine die vom Execution Server benutzt wird. Der Datenbankbenutzer benötigt die Rechte um Tabellen zu erstellen, ändern und löschen zu können.

1.6. Execution Server Home

Das Execution Server Home ist ein Ordner der für die Speicherung von Daten und Konfigurationen des Execution Server benutzt wird. Der Ordnername ist normalerweise `visualrules-executionserver` und befindet sich im Benutzerverzeichnis des Betriebssystembenutzers, der den Execution Server startet. Es ist auch möglich einen anderen Ordner anzugeben, wie in Abschnitt 1.5, „Konfiguration des Execution Server“ beschrieben wird. Bei Bedarf wird der Ordner automatisch angelegt.

Tabelle 1.3. Execution Server Home Inhalt

vrdbartifacts	Ordner der die Datenbank enthält, wenn die eingebettete Apache Derby verwendet wird
logs	Ordner der die Protokoll Dateien des Execution Servers enthält
derby.log	Protokoll Datei der Apache Derby
vr_logging.config	Konfigurations Datei für die Laufzeit Protokollierung

1.7. Laufzeitprotokollierung

Ab der Version 5.4 wird die Laufzeitprotokollierung vom Execution Server automatisch eingerichtet (Bitte beachten: Log4j wird nicht mehr verwendet). Beim Startvorgang wird die Laufzeitprotokollierung initialisiert und Nachrichten werden in eine Protokolldatei (auch log genannt) geschrieben, welche sich im `logs` Ordner im Execution Server Home befindet. Eine Protokoll Datei wird bis zu 3 Megabyte (MB) groß und wird dann archiviert. Es gibt maximal drei Archive die eine Nummer im Dateinamen tragen. Zum Beispiel ist "3" das dritte und somit älteste Archiv. Der Execution Server schreibt zwei unterschiedliche Protokolle:

- `executionserver.log` enthält die Nachrichten vom Execution Server und verwendeter Komponenten von Drittanbietern
- `LogAction.log` enthält Nachrichten die in Rule Services von Aktionen vom Typ *"Log Eintrag schreiben"* geschrieben werden

Die **Kategorie** Eigenschaft von Aktionen vom Typ *"Log Eintrag schreiben"* wird nicht unterstützt. Wird dies verwendet, so werden die Nachrichten nicht in die `LogAction.log` Datei geschrieben.

Die Protokoll Datei kann in der Webkonsole eingesehen werden, wo es auch möglich ist, Einstellungen für die Laufzeitprotokollierung vorzunehmen. Mehr Informationen dazu finden sich in Abschnitt 3.5.2, „Konfiguration und Anzeige von Nachrichten der Laufzeitprotokollierung“.

Sollte der Fall eintreten, dass der Execution Server nicht starten kann oder es aus anderen Gründen nicht möglich ist, auf die Webkonsole zuzugreifen, dann kann die Laufzeitprotokollierung in der `vr_logging.config` Datei im Execution Server Home konfiguriert werden.

Das sollte nur in Ausnahmesituationen benutzt werden und ist nicht für die Verwendung im Normalbetrieb gedacht. Diese Einstellung ist nur möglich, wenn der Execution Server nicht läuft. Die Konfigurationsdatei wird nur beim Startvorgang eingelesen und Einstellungen in der Webkonsole überschreiben Werte in der Datei.

Bevor die Protokollierung in der Konfigurations Datei vorgenommen werden kann, muss sichergestellt sein, dass der Execution Server nicht läuft. Dann kann die `vr_logging.config` Datei in einem Texteditor geöffnet werden. Die Datei enthält XML Elemente names `loggers` und `logger`. Diese haben Attribute names `level` welche den Protokollierungsgrad (log level) spezifizieren, die ähnlich zu den in Simple Logging Facade for Java (SLF4J) sind. Der Wert kann geändert werden zu: `TRACE`, `DEBUG`, `INFO`, `WARN` oder `ERROR`. Für die Fehlersuche eignet sich `DEBUG`. Nach dem Speichern der Änderung, sollte die Protokolldatei geleert werden, damit Fehler leichter zu finden sind. Beim anschließenden Start des Execution Servers wird die neue Konfiguration angewendet und Nachrichten mit dem eingestellten Protokollierungsgrad sollten in der Protokolldatei zu finden sein.

Kapitel 2. Erstellung und Bereitstellung von Rule Services

2.1. Konzepte

2.1.1. Rule Service

Der Visual Rules Execution Server ist eine Ausführungsumgebung für Regeln. Regeln werden mit dem Visual Rules Modeler erstellt und können dann auf dem Visual Rules Execution Server bereitgestellt werden. Einmal bereitgestellt, stehen die Regeln für jegliche andere Applikationen oder Systeme, welche die Regeln aufrufen möchten, zur Verfügung. Der Visual Rules Execution Server stellt eine Web Service Schnittstelle zur Verfügung, um Regeln aufzurufen. Auf diese Weise wird jede Regel zu einem separaten Service - ein sogenannter Rule Service.

Die Regeln für einen Rule Service müssen für die Anwendung in eine Regelbibliothek gepackt werden. Diese Bibliothek und die entsprechenden Services haben Versionsnummern, um eindeutig während des Lebenszyklusses eines Regelprojektes identifiziert werden zu können. Mehrere Versionen des gleichen Rule Service können im Visual Rules Execution Server bereitgestellt und zur gleichen Zeit ausgeführt werden.

Ein Rule Service wird durch eine WSDL Datei (Web Service Description Language) beschrieben, die automatisch durch Visual Rules während der Erstellung der Regelbibliothek generiert wird, um dann nach der Bereitstellung verfügbar zu sein. Auf die WSDL Dateien aller eingesetzten Rule Services kann über die Execution Server Webkonsole zugegriffen werden.

Weiterführende Konzepte.

- Abschnitt 2.1.2, „Regelbibliothek“
- Abschnitt 2.1.3, „Versionen von Regelbibliothek und Rule Service“

Weiterführende Arbeitsschritte.

- Abschnitt 2.2.1, „Regeln festlegen, die als Rule Service exportiert werden sollen“
- Abschnitt 3.2.5, „Anzeige der WSDL-Datei eines Rule Service“

2.1.2. Regelbibliothek

Um einen Rule Service im Visual Rules Execution Server zu verwenden, muss das entsprechende Regelprojekt in eine sogenannte Regelbibliothek gepackt werden. Regelbibliotheken sind JAR Dateien, die den generierten Regelcode für alle Regelmodelle, die im Regelprojekt enthalten sind, beinhalten (siehe Abbildung 2.1, „Regelprojekt, Regelbibliothek und Rule Service“).

Die Regelbibliothek beinhaltet somit die WSDL Datei (und XML Schemas) für das Regelmodell, das als Web Service exportiert wird (es kann nur ein Regelmodell eines Regelprojekts als Rule Service exportiert werden, die anderen Regelmodelle bleiben intern). Die WSDL und XML Schema Dateien werden automatisch von Visual Rules erstellt, wenn es eine Regelbibliothek paketiert.

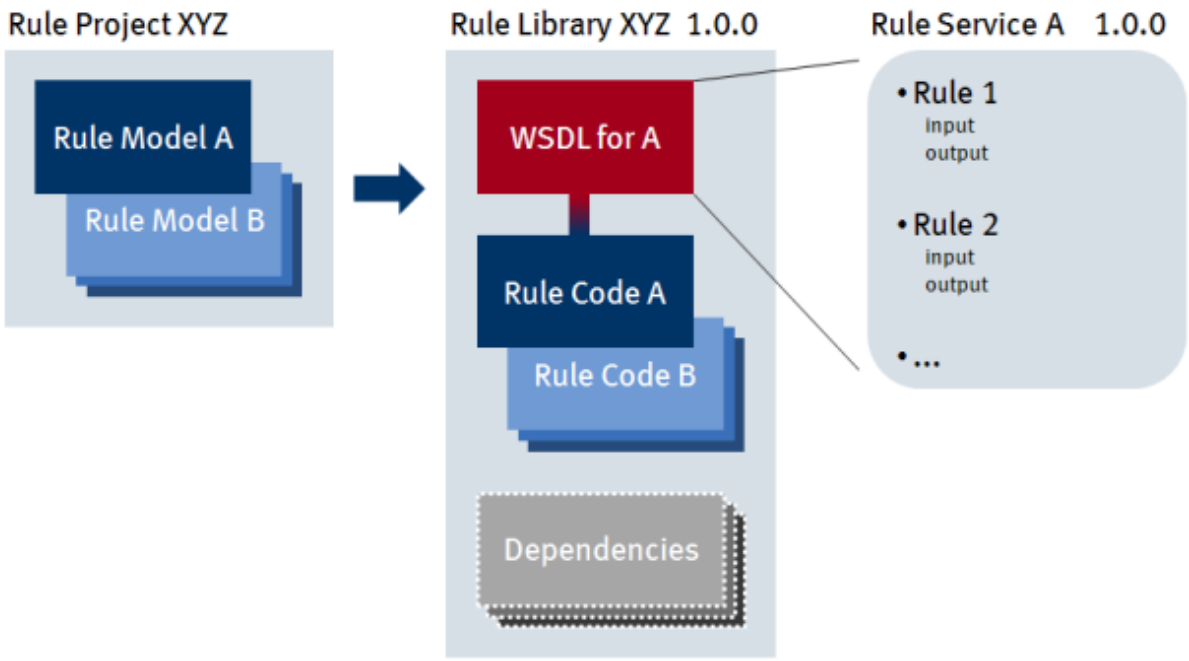


Abbildung 2.1. Regelprojekt, Regelbibliothek und Rule Service

Der Visual Rules Modeler paketiert Regelbibliotheken, so dass sie den Regelcode von allen Regelmodellen im Projekt UND den Regelcode von allen abhängigen Regelmodellen enthalten. Die Regelbiblothek beinhält auch alle Klassen von anderen Bibliotheken und Java Projekten auf dem Erstellungspfad des Projekts. Auf diese Weise enthält die resultierende Regelbibliothek alle benötigten Klassen für die Regeln zur Ausführung. Beachten Sie jedoch, dass die Visual Rules Laufzeit Bibliotheken nicht in die Regelbibliothek gepackt werden. Sie werden zur Laufzeit automatisch zum Klassenpfad des Execution Servers hinzugefügt.

Weiterführende Konzepte.

- Abschnitt 2.1.1, „Rule Service“
- Abschnitt 2.1.3, „Versionen von Regelbibliothek und Rule Service“

Weiterführende Arbeitsschritte.

- Abschnitt 2.2.1, „Regeln festlegen, die als Rule Service exportiert werden sollen“

2.1.3. Versionen von Regelbibliothek und Rule Service

Eine Regelbibliothek hat immer eine Versionsnummer. Diese Versionsnummer wird verwendet, um eine spezifische Regelbibliothek und Rule Service während der Bereitstellung und der Ausführung von Regeln zu identifizieren.

Die Versionsnummer besteht aus drei individuellen Nummern, die durch Punkte getrennt sind, d.h. `1.0.0` oder `12.4.11`. Versionsnummern werden verwendet, um eindeutig eine spezifische Version einer Regelbibliothek zu identifizieren. Dies ist notwendig, weil während des Lebenszyklusses eines Regelprojekts viele verschiedene Versionen für die Ausführungsumgebung zur Verfügung gestellt werden können. So können mehrere Versionen des gleichen Regelprojekts zur gleichen Zeit eingesetzt werden. Durch Verwendung der Versionsnummer, kann ein Aufrufer genau spezifizieren, welche Version der Regel aufgerufen werden soll.

Die drei Komponenten einer Versionsnummer werden "major", "minor" und "micro" genannt. Höhere Nummern werden als "neuere" Versionen angesehen. Die micro und/oder minor Komponenten können weggelassen werden und werden dann als `0` behandelt, beispielsweise entspricht die Versionsnummer `2.6` der Nummer `2.6.0`, und `1` entspricht `1.0` beziehungsweise `1.0.0`.

Optional kann nach der "micro" Komponente, eine weitere Komponente angegeben werden, die "qualifier" genannt wird. Der Qualifier wird durch einen Bindestrich (-) getrennt und kann zusätzlich zur Identifikation verwendet werden, beispielsweise kann dies ein Zeitstempel sein. Der Qualifier kann nicht nur Nummern enthalten, sondern

auch Buchstaben (`A-Z`, `a-z`), Unterstriche (_) und Bindestriche (-). Qualifier werden lexikographisch geordnet, um zu bestimmen, welche Version "neuer" ist.

Unter Umständen kann es dazu kommen, dass zwei Rule Services die gleiche Version haben, aber nur einer der beiden einen "qualifier" hat. Wenn in so einem Fall der Rule Service ohne eine spezifizierte Version aufgerufen wird, dann wird der Rule Service ohne "qualifier" als "neuer" angesehen.

Weiterführende Konzepte.

- Abschnitt 2.1.1, „Rule Service“
- Abschnitt 2.1.2, „Regelbibliothek“

Weiterführende Arbeitsschritte.

- Abschnitt 2.2.4, „Bereitstellen von Regelprojekten vom Visual Rules Modeler“

2.1.4. Visual Rules Archiv

Der Execution Server unterstützt auch das Visual Rules Archiv, welches neben dem Rule Service auch noch alle weiteren Bibliotheken, die zur Ausführung eines Rule Services notwendig sind, und eine Abhängigkeitsbeschreibung enthält.

In dieser Form können Rule Services von einem Visual Rules Execution Server herunter- bzw. hochgeladen werden (z.B. um Rule Services von einer Test- auf eine Produktionsumgebung zu übertragen).

Die zugehörige Archivdatei besitzt die Endung VRA.

Weiterführende Konzepte.

- Abschnitt 2.1.1, „Rule Service“
- Abschnitt 2.1.2, „Regelbibliothek“

Weiterführende Arbeitsschritte.

- Abschnitt 3.2.6, „Herunterladen eines Rule Service“
- Abschnitt 3.2.3, „Hinzufügen eines Rule Service mittels Visual Rules Archiv“

2.1.5. Rule Service Einstellungen

Rule Services können zusätzliche Einstellungen haben, die im folgenden beschrieben sind.

2.1.5.1. Aktiv

Ein Rule Service ist standardmäßig aktiv und kann somit aufgerufen werden. Es kann allerdings auch Situationen geben, in denen Rule Services zwar bereitgestellt sind, aber nicht aufgerufen werden sollen. Für solche Zwecke kann ein Rule Service deaktiviert werden, wodurch dieser nicht mehr aufgerufen werden kann. Für Aufrufer ist es dann nicht unterscheidbar, ob der Rule Service gelöscht wurde oder nur deaktiviert wurde.

2.1.5.2. Gültig von und Gültig bis

Diese Werte werden vom Standard-Metadata Mapper in einer generischen Rule Service Anfrage verwendet. Beim Einsatz eines benutzerdefinierten Metadata Mappers, wie in Abschnitt 6.2.1.3, „Benutzerdefinierter Metadata Mapper“ beschrieben, kann das ebenfalls genutzt werden.

2.1.5.3. Name der aktiven Konfiguration (Active Configuration Name)

Im Visual Rules Modeler können mehrere Konfigurationen für die Regelausführung auf einem Regelmodell definiert werden. Diese können beispielsweise in der Visual Rules Rule Execution API verwendet werden, um Implementierungen von Aktionen auszutauschen. Mehr Informationen zu diesem Thema finden sich im *Java Integration Handbuch*.

Der Execution Server bietet Aufrufern ebenfalls die Möglichkeit, den Namen der zu verwendenden Konfiguration anzugeben. Im Englischen und den technischen Schnittstellen wird hierfür der Begriff `active configuration name` verwendet. Zu beachten ist hierbei, dass diese Angabe im Aufruf in älteren Versionen der Visual Rules Rule Execution API und im Execution Server als `binding` bezeichnet wird.

Jeder Rule Service kann konfiguriert werden, einen speziellen Namen als Standardwert für die aktive Konfiguration zu verwenden. Dieser kann anders sein, als derjenige, der normalerweise in der Visual Rules Rule Execution API verwendet wird. Dies wird dann als `active configuration name` verwendet, wenn ein Rule Service ausgeführt wird. Ein Aufrufer kann ein `active configuration name` auch in der Rule Service Anfrage angeben (für Details siehe Abschnitt 4.1.1, „Format der Rule Service Anfrage“). Die Einstellung in der Anfrage übertrumpft in diesem Fall immer den Wert der am Rule Service eingestellt wurde.

Rule Services, die von Visual Rules 4 migriert wurden, unterstützen das Setzen eines `active configuration name` als Standardwert nicht.

2.1.5.4. Statistiklevel

Die Visual Rules Rule Execution API erlaubt es, den Statistiklevel der aufgezeichneten Statistiken einzustellen. Mehr zu diesem Thema findet sich im *Java Integration Guide*.

Diese Funktionalität wird als Aufzeichnung von Ausführungen auch vom Execution Server unterstützt. Der Statistiklevel eines Rule Service kann eingestellt werden, welcher dann als Standardwert verwendet wird. Aufrufer können einen Statistiklevel auch in der Rule Service Anfrage spezifizieren (siehe Abschnitt 4.1.1, „Format der Rule Service Anfrage“), welcher dann statt der Einstellung des Rule Service verwendet wird.

Zusätzlich zu den Leveln in der Visual Rules Rule Execution API, hat der Execution Server den Level `QUIET`, mit dem die Aufzeichnung der Statistik abgeschalten werden kann. Ein Rule Service kann auch so eingestellt werden, dass keine Ausführungen aufgezeichnet werden, indem der Level auf `SWITCHED_OFF` gestellt wird.

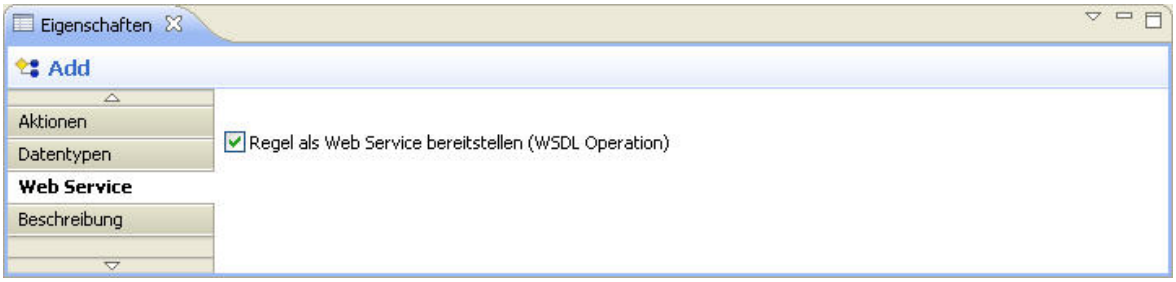
Rule Services, die von Visual Rules 4 migriert wurden, unterstützen es nicht, einen Standardwert für den Statistiklevel einzustellen. Jedoch lässt sich die Aufzeichnung von Ausführungen durch Einstellen des Levels `SWITCHED_OFF` ausschalten.

2.2. Arbeitsschritte

2.2.1. Regeln festlegen, die als Rule Service exportiert werden sollen

Nur exportierte Regeln können durch einen Rule Service Client aufgerufen werden. Somit muss mindestens eine Regel eines Regelmodells exportiert werden, um einen Rule Service zu erstellen. Um zu spezifizieren, welche Regeln exportiert werden sollen, führen Sie folgendes durch:

1. Im Projekt Explorer oder Regel Explorer wählen Sie eine Regel zum Exportieren aus.
2. Gehen Sie auf die Seite Web Service in der Sicht Eigenschaften.
3. Aktivieren Sie das Kästchen Regel als Web Service bereitstellen (WSDL operation).



Weiterführende Konzepte.

- Abschnitt 2.1.1, „Rule Service“

- Abschnitt 2.1.2, „Regelbibliothek“

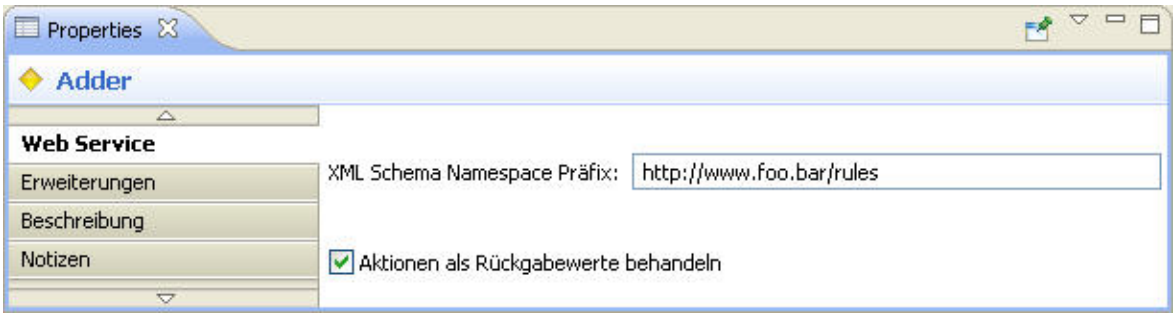
2.2.2. XML Namespace für Rule Services definieren

Die generierte WSDL und die XML Schemata für einen Rule Service verwenden einen spezifischen XML Namespace. Wenn Sie die Namespace URI für einen Rule Service anpassen möchten, führen Sie folgende Schritte durch:

1. Im Projekt Explorer oder Regel Explorer wählen Sie das entsprechende Regelmodell aus. Dies ist das Regelmodell, welches für den Rule Service exportiert wird.
2. Gehen Sie auf die Seite Web Service in der Sicht Eigenschaften.
3. Geben Sie den XML Schema Namespace Präfix ein. Dies muss eine gültige URI sein (typischerweise wird http:// als Schema verwendet).

   Der vollständige Namespace URI im WSDL und XML Schema wird später aus diesem Präfix gebildet, gefolgt vom Segment `/vrpath`, gefolgt vom Pfad der spezifischen Regel.

Ist kein Präfix angegeben, wird `http://www.visual-rules.com` als Standardwert verwendet.



Weiterführende Konzepte.

- Abschnitt 2.1.1, „Rule Service“
- Abschnitt 2.1.2, „Regelbibliothek“
- Abschnitt 4.1.1, „Format der Rule Service Anfrage“
- Abschnitt 4.1.2, „Format der Rule Service Antwort“

2.2.3. Einstellen von Aktionen als Rückgabewert

Aktionen sind ein Teil des Ergebnisses eines Regelaufrufs. Somit sind sie auch Teil der Rückgabe der generierten XML Schema Dateien für einen Rule Service. Es ist möglich einzustellen, ob Aktionen als Rückgabewert behandelt werden. Das führt dann zu unterschiedlichen XML Schema Definitionen für einen Web Service. Um dies einzustellen, führen Sie folgende Schritte durch:

1. Im Projekt Explorer oder Regel Explorer wählen Sie das entsprechende Regelmodell aus. Dies ist das Regelmodell, welches für den Rule Service exportiert wird.
2. Gehen Sie auf die Seite Web Service in der Sicht Eigenschaften.
3. Durch Einstellen der Checkbox kann das Verhalten umgestellt werden. Das hat einen Effekt auf die Elemente, welche in der Antwort zurückgeliefert werden.

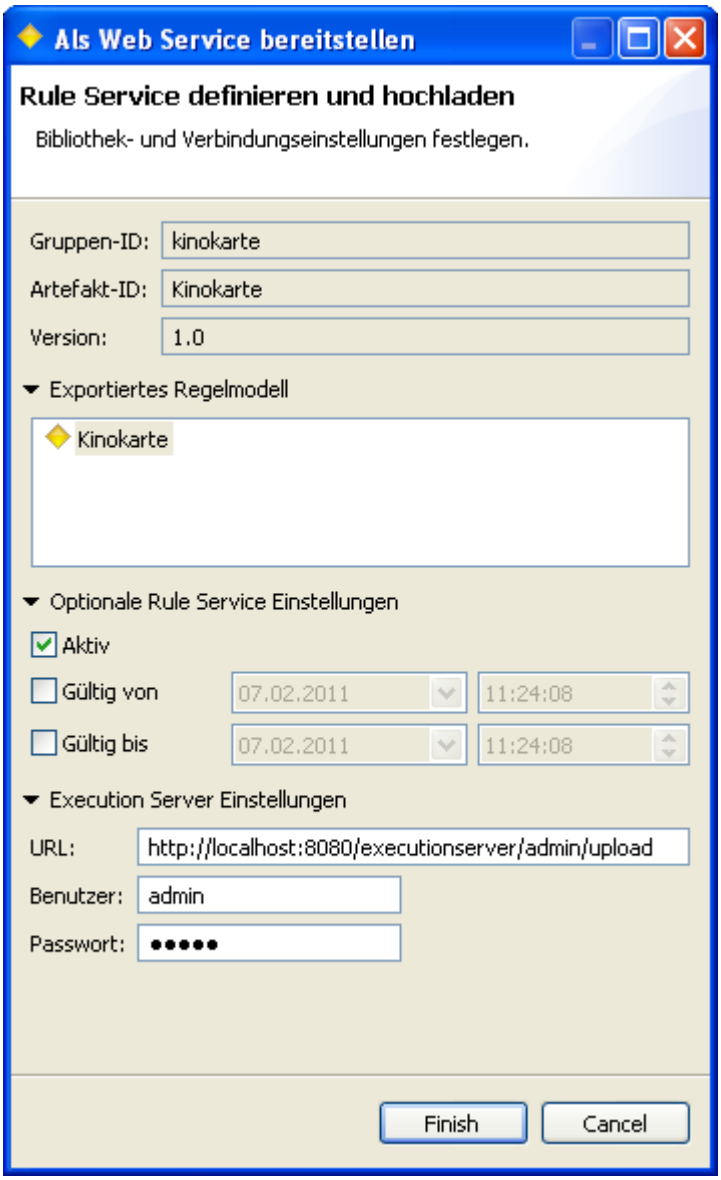
2.2.4. Bereitstellen von Regelprojekten vom Visual Rules Modeler

Regelprojekte können direkt im Visual Rules Modeler paketiert und auf dem Execution Server bereitgestellt werden. Diese Funktionalität ist auch in Visual Rules Builder (separat verfügbar) enthalten.

Für die Bereitstellung muss der Benutzer über die erforderliche Berechtigung verfügen. Siehe auch Abschnitt 5.1, „Verfügbare Rollen und Berechtigungen“.

1. Im Regel Explorer oder Projekt Explorer machen Sie einen Rechtsklick auf das Regelprojekt, das bereitgestellt werden soll.

2. Wählen Sie den Visual Rules > Als Web Service bereitstellen Menüeintrag. Der Assistent für die Bereitstellung erscheint.



3. Die Artefakt-ID, Gruppen-ID und Version sind auf der `ruleproject.vr` definiert und werden zur Identifizierung des Artefakts (JAR) verwendet. Diese Version gilt für die Regelbibliothek selbst und den Rule Service, der bereitgestellt wird.

4. Wenn das Regelprojekt mehrere Regelmodelle enthält, müssen Sie auswählen, welches Regelmodell exportiert werden soll. Erweitern Sie den Abschnitt Exportiertes Regelmodell und wählen Sie das Regelmodell zum Exportieren aus. Dies ist nicht notwendig, wenn das Regelprojekt nur ein einziges Regelmodell hat.

5. Im Abschnitt Optionale Rule Service Einstellungen können Werte für die Gültigkeitsdauer eingestellt und der Rule Service aktiviert bzw. deaktiviert werden.

6. Erweitern Sie den Abschnitt Execution Server Einstellungen und prüfen Sie die Werte für die Execution Server URL, den Benutzer und das Passwort.

Die URL endet immer mit `/admin/upload`. Der Host Name und der Kontext Name hängen von Ihrer Installation des Execution Server ab.

In der Abbildung oben läuft der Execution Server auf dem lokalen Rechner mit dem Port 8080 und der Webapplikations-Kontext wurde `executionserver` genannt.

7. Klicken Sie auf Fertig stellen.

8. Visual Rules Modeler wird nun die Abhängigkeiten des Regelprojekts analysieren, den Regelcode generieren und diesen in eine Regelbibliothek mit allen anderen Abhängigkeiten auf dem Erstellungspfad der involvierten Projekte packen. Die Regelbibliothek wird dann zum Execution Server hochgeladen.

Weiterführende Konzepte.

- Abschnitt 2.1.1, „Rule Service“
- Abschnitt 2.1.2, „Regelbibliothek“

Weiterführende Arbeitsschritte.

- Abschnitt 2.2.1, „Regeln festlegen, die als Rule Service exportiert werden sollen“

Kapitel 3. Arbeiten mit der Webkonsole

Die folgenden Abschnitte enthalten Beschreibungen verschiedener Aufgaben, die mit der Webkonsole ausgeführt werden können, z.B. bereitgestellte Rule Services verwalten.

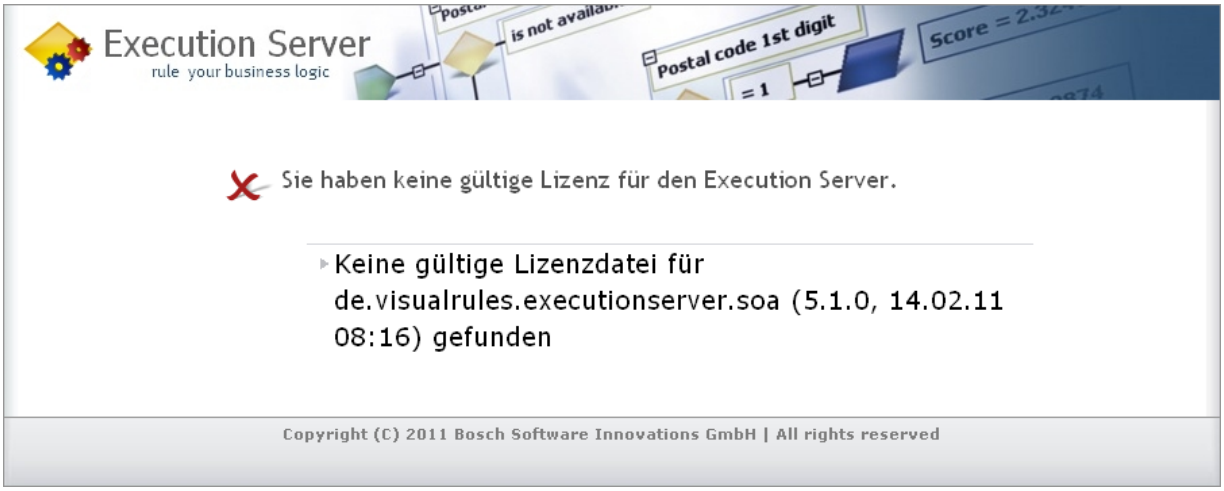
3.1. Aufruf der Webkonsole

Nach erfolgreicher Installation des Execution Servers kann die Webkonsole in einem Web Browser aufgerufen werden. Sie ist unter der Adresse `http://<server>:<port>/<context-name>` erreichbar, wobei die Werte meist durch den Application Server vorgegeben sind.

Bei einer Bereitstellung auf einem `Apache Tomcat`, unter Verwendung dessen Standardkonfiguration, lautet die Adresse beispielsweise `http://localhost:8080/executionserver-5.x.y` (wobei x und y Platzhalter für die Versionsnummern aus dem Dateinamen der WAR Datei des Execution Servers darstellen).

Es erscheint die folgende Startseite zur Anmeldung:

Wenn beim Start des Execution Servers keine gültige Lizenz zur Verfügung stand, erscheint die Startseite mit einer Fehlermeldung:


Execution Server
rule your business logic
Sie haben keine gültige Lizenz für den Execution Server.
Keine gültige Lizenzdatei für de.visualrules.executionserver.soa (5.1.0, 14.02.11 08:16) gefunden
Copyright (C) 2011 Bosch Software Innovations GmbH | All rights reserved

Weiterführende Aufgaben.

- Abschnitt 1.4, „Installation der Lizenz“

3.1.1. Einstellung einer bestimmten Sprache

Die Seiten der Webkonsole werden in der Sprache angezeigt, die Sie als bevorzugte Sprache in Ihrem Web Browser eingestellt haben. Sollte diese Sprache von der Webkonsole noch nicht unterstützt werden, dann werden die Seiten in *Englisch* angezeigt.

Es ist möglich, explizit die *deutsche* bzw. *englische* Sprache für die Webkonsole einzustellen:

1. Öffnen Sie eine Seite der Webkonsole in Ihrem Web Browser.

2. Bewegen Sie die Maus über das Flaggensymbol im rechten Teil der Menüleiste.

3. Klicken Sie im Kontextmenü auf die Flagge , wenn Sie die *deutsche* Sprache einstellen wollen bzw. auf die Flagge , wenn Sie die *englische* Sprache wünschen.

3.2. Verwaltung bereitgestellter Rule Services

In der Sicht *Rule Services* des Visual Rules Execution Servers können Sie bereitgestellte Rule Services verwalten.

Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services, um diese Sicht zu öffnen.

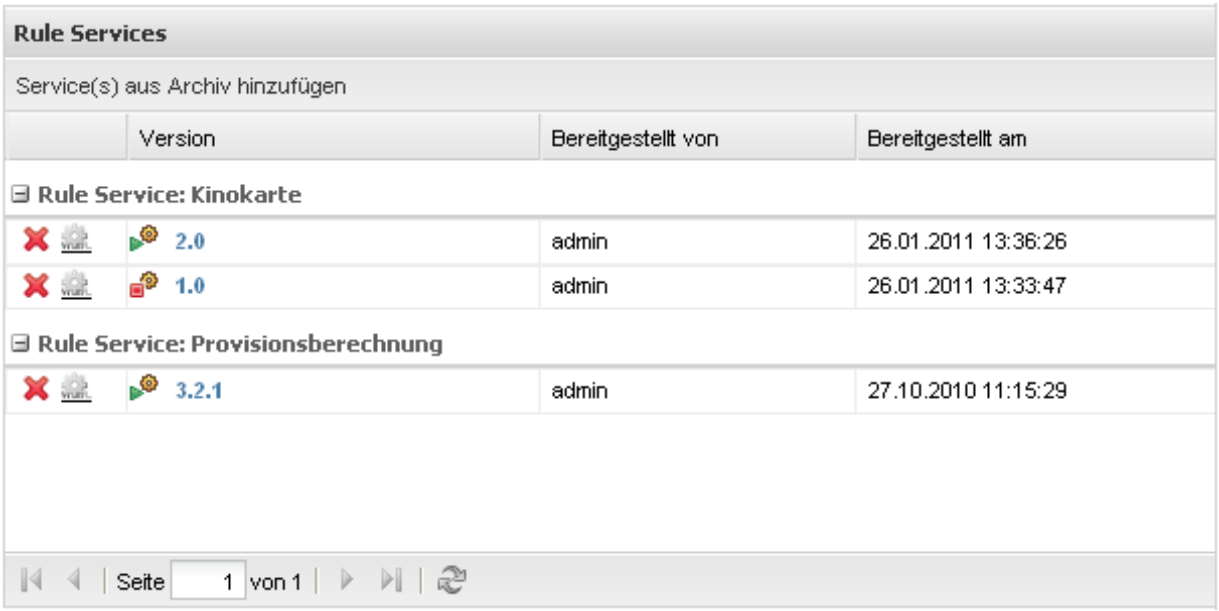
Hier haben Sie folgende Möglichkeiten:

- Abschnitt 3.2.1, „Anzeige bereitgestellter Rule Services“
- Abschnitt 3.2.2, „Filterung angezeigter Rule Services“
- Abschnitt 3.2.3, „Hinzufügen eines Rule Service mittels Visual Rules Archiv“
- Abschnitt 3.2.4, „Löschen eines bereitgestellten Rule Service“
- Abschnitt 3.2.5, „Anzeige der WSDL-Datei eines Rule Service“
- Abschnitt 3.2.6, „Herunterladen eines Rule Service“
- Abschnitt 3.2.7, „Anzeige der Eigenschaften und Änderung der Einstellungen eines Rule Service“
- Abschnitt 3.2.8, „Verwaltung der Metadaten eines Rule Service“
- Abschnitt 3.2.9, „Anzeige der Ausführungen eines Rule Service“
- Abschnitt 3.2.10, „Anzeige der erforderlichen Bibliotheken eines Rule Service“

3.2.1. Anzeige bereitgestellter Rule Services

Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services aus, um die Sicht *Rule Services* zu öffnen.

Die geöffnete Übersichtsseite enthält den Abschnitt "Rule Services", in dem alle auf dem Visual Rules Execution Server bereitgestellten Rule Services angezeigt werden:



Standardmäßig werden folgende Eigenschaften der Rule Services angezeigt:

- Name
- Aktiviert oder deaktiviert
- Version
- Bereitgestellt von
- Bereitgestellt am

Sie können das Layout der angezeigten Eigenschaften beeinflussen, indem Sie beispielsweise

- bestimmte Eigenschaften ein- bzw. ausblenden. Neben den standardmäßig angezeigten Eigenschaften gibt es noch weitere Eigenschaften wie *Artefakt-ID*, *Gruppen-ID* und Metadaten, die eingeblendet werden können. Besitzt ein Rule Service die zur Anzeige konfigurierten Metadaten nicht, dann werden die betreffenden Spalten in der Tabelle leer gelassen. (siehe Abschnitt 3.6.1, „Ein- und Ausblenden von Spalten“)
- zugehörige Spalten verschieben (siehe Abschnitt 3.6.2, „Verschieben einer Spalte“)
- das Sortierkriterium und die Sortierreihenfolge ändern (siehe Abschnitt 3.6.3, „Verändern des Sortierkriteriums und der Sortierreihenfolge“)

Rule Services mit identischem Namen werden gruppiert, d.h. sie werden immer als Einheit direkt untereinander aufgelistet. Initial werden alle Rule Services einer Gruppe angezeigt. Möchten Sie die einzelnen Rule Services einer Gruppe nicht betrachten, so können Sie diese durch Klicken auf die zur Gruppe gehörenden Schaltfläche ausblenden (und durch Klicken auf auch wieder anzeigen).

Weiterführende Konzepte.

- Abschnitt 2.1.1, „Rule Service“

Weiterführende Aufgaben.

- Abschnitt 3.2.2, „Filterung angezeigter Rule Services“
- Abschnitt 3.2.3, „Hinzufügen eines Rule Service mittels Visual Rules Archiv“
- Abschnitt 3.2.4, „Löschen eines bereitgestellten Rule Service“
- Abschnitt 3.2.5, „Anzeige der WSDL-Datei eines Rule Service“
- Abschnitt 3.6, „Konfiguration der Anzeige von Tabelleninhalten“

3.2.2. Filterung angezeigter Rule Services

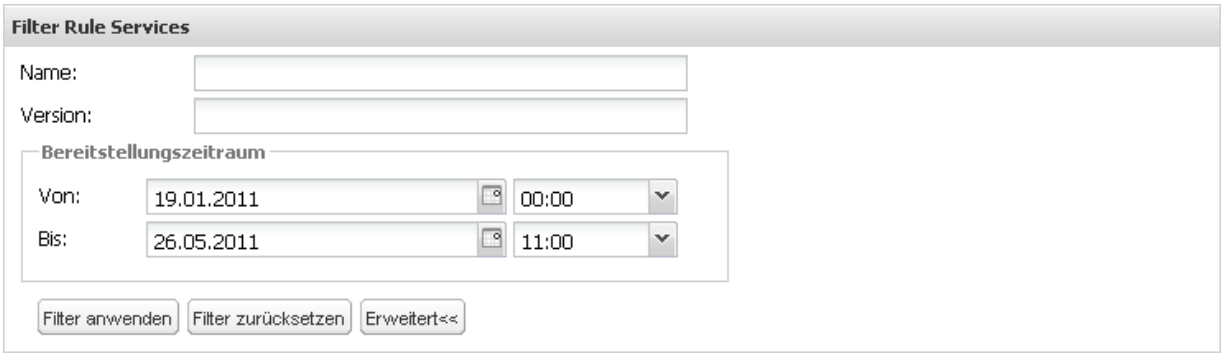
Auf der Übersichtsseite der Sicht *Rule Services* befindet sich der Abschnitt "Filter Rule Services", in dem Sie Filterkriterien spezifizieren können, um nur Rule Services, die von Interesse sind, anzuzeigen:

In diesem Abschnitt können Sie den Namen des Rule Service als Filterkriterium angeben.

Sie haben die Möglichkeit, weitere Filterkriterien für die Anzeige der Rule Services zu spezifizieren:

- Version
- Bereitstellungszeitraum

Dazu steht ein erweiterter Eingabedialog zur Verfügung:



Mehr Informationen zur Eingabe der Filterkriterien und der Anwendung des Filters finden Sie in Abschnitt 3.7, „Filterung angezeigter Objekte“.

3.2.3. Hinzufügen eines Rule Service mittels Visual Rules Archiv

Der Execution Server unterstützt das Hochladen eines *Visual Rules Archivs*. Dieses enthält neben dem Rule Service selbst auch alle weiteren, zur Serviceausführung erforderlichen Bibliotheken.

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services aus.

2. Klicken Sie im Abschnitt "Rule Services" auf die Schaltfläche Service(s) aus Archiv hinzufügen.

   Diese Schaltfläche ist nur dann aktiviert, wenn Sie ausreichende Berechtigungen haben. Der Standardbenutzer `admin` hat diese Erlaubnis. Siehe auch Abschnitt 5.1, „Verfügbare Rollen und Berechtigungen“.

   Der folgende Dialog erscheint:

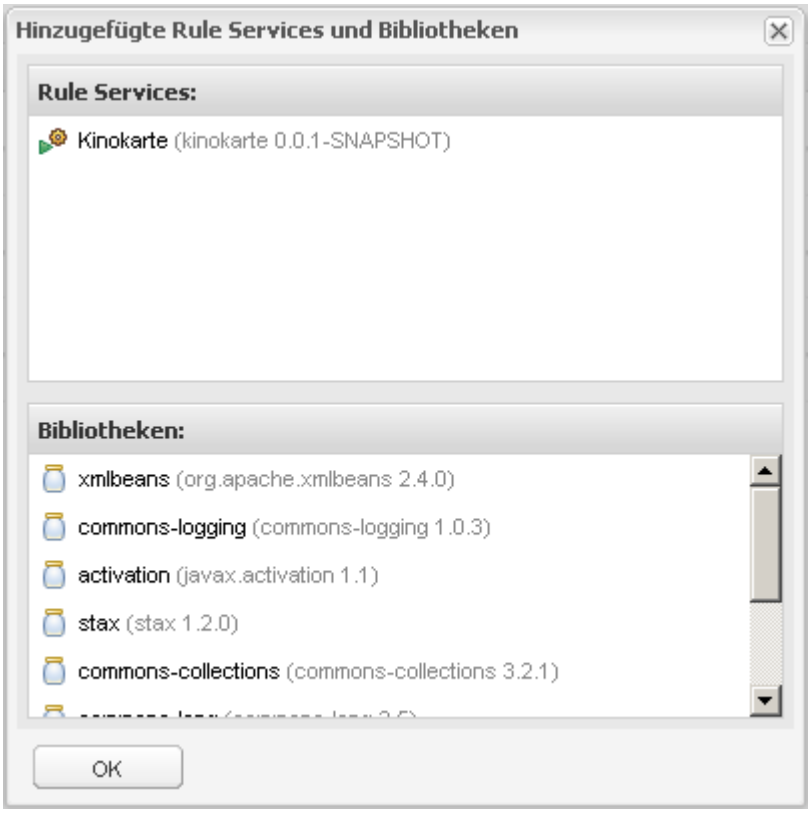
3. Klicken Sie auf die Schaltfläche Durchsuchen....

   Es öffnet sich ein Dialog zum Hochladen des *Visual Rules Archivs*. Er ermöglicht Ihnen die Spezifikation einer *Visual Rules Archiv*-Datei (mit der Endung *.vra*). Das Aussehen des Dialoges ist abhängig von Ihrem Web Browser.

   Der Pfad der ausgewählten Datei wird anschließend im Eingabefeld angezeigt.

4. Klicken Sie auf die Schaltfläche Hochladen, um die spezifizierte Datei hochzuladen.

   Wenn die Operation abgeschlossen ist, wird eine Statusmeldung angezeigt. Zudem erscheint nach erfolgreicher Operation ein Dialog, in dem alle hinzugefügten Rule Services und Bibliotheken aufgelistet sind:



Weiterführende Konzepte.

- Abschnitt 2.1.4, „Visual Rules Archiv“

3.2.4. Löschen eines bereitgestellten Rule Service

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services aus.
2. Klicken Sie im Abschnitt "Rule Services" auf das Symbol des Rule Service, den Sie löschen möchten.

   Dieses Symbol ist nur verfügbar, wenn Sie ausreichende Berechtigungen haben. Der Standardbenutzer `admin` hat diese Erlaubnis. Siehe auch Abschnitt 5.1, „Verfügbare Rollen und Berechtigungen“.
3. Es öffnet sich ein Dialog, in dem Sie bestätigen können, dass Sie die Aktion wirklich durchführen möchten.

   Klicken Sie zur Bestätigung auf die Schaltfläche

.

3.2.5. Anzeige der WSDL-Datei eines Rule Service

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services aus.
2. Klicken Sie im Abschnitt "Rule Services" auf das Symbol des Rule Service, dessen WSDL-Datei angezeigt werden soll.
3. Die WSDL-Datei wird im Web Browser geöffnet.

Falls Sie die Detailseite eines Rule Service geöffnet haben, können Sie auch dort die Anzeige der WSDL-Datei veranlassen:

1. Klicken Sie auf den Link

im oberen Bereich der Detailseite.

2. Die WSDL-Datei wird im Web Browser geöffnet.

Die URL für die WSDL lautet:

```
http://<server>:<port>/<context-name>/services/<rule-model>@<version>/<rule-model>.wsdl
```

Das WSDL importiert zusätzliche XML Schema-Dateien, welche wiederum andere Schema-Dateien importieren können. Alle diese Ressourcen können vom Execution Server heruntergeladen werden.

Sie können diese WSDL für jeden Web Service Client verwenden, um ein Regelmodell aufzurufen.

Weiterführende Konzepte.

- Abschnitt 4.1.5, „Abbildung des Regelmodells auf WSDL“

3.2.6. Herunterladen eines Rule Service

Sie können einen Rule Service als *Regelbibliothek* oder als *Visual Rules Archiv* herunterladen. Ein *Visual Rules Archiv* enthält neben dem Rule Service auch noch alle weiteren, zur Serviceausführung erforderlichen Bibliotheken und kann auf einem anderen Visual Rules Execution Server hochgeladen werden.

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services aus.
2. Öffnen Sie die Detailseite des Rule Service, den Sie herunterladen wollen, indem Sie im Abschnitt "Rule Services" auf den zugehörigen Link in der Versionsspalte klicken.
3. Klicken Sie im oberen Bereich der Detailseite auf den Link Regelbibliothek herunterladen, wenn Sie nur die *Regelbibliothek* herunterladen wollen,

   oder

   auf den Link Visual Rules Archiv herunterladen, wenn Sie das *Visual Rules Archiv* herunterladen wollen.
4. Es öffnet sich ein Dialog zum Herunterladen der Datei. Er ermöglicht Ihnen das Speichern der Datei. Das Aussehen des Dialoges ist abhängig von Ihrem Web Browser.

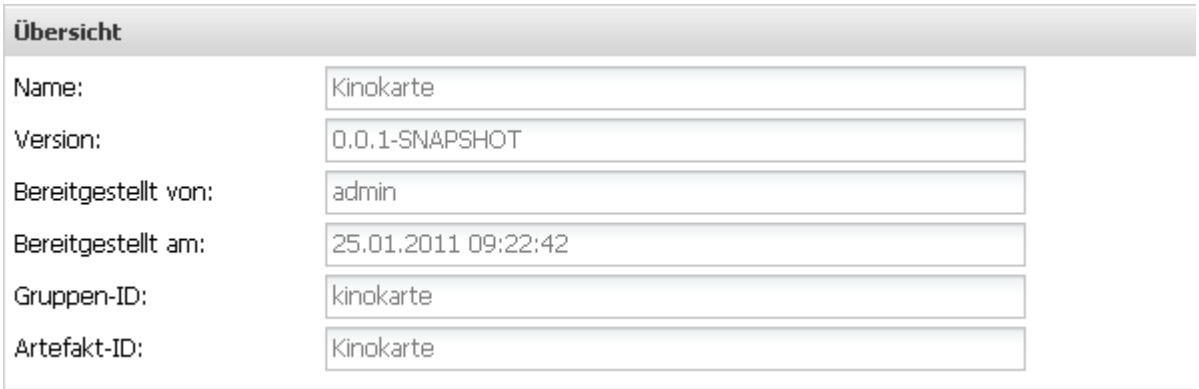
Weiterführende Konzepte.

- Abschnitt 2.1.2, „Regelbibliothek“
- Abschnitt 2.1.4, „Visual Rules Archiv“

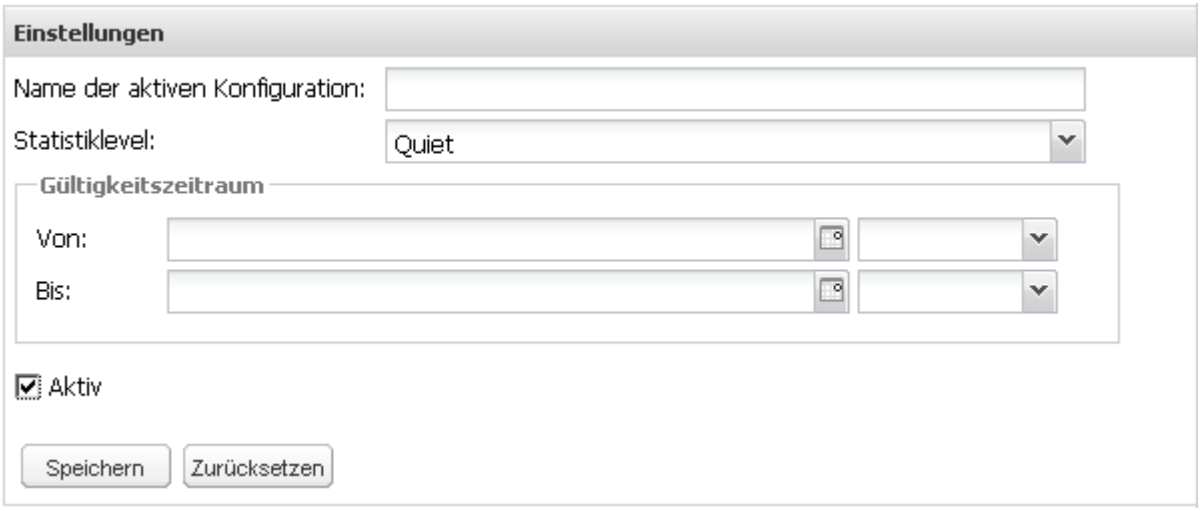
3.2.7. Anzeige der Eigenschaften und Änderung der Einstellungen eines Rule Service

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services aus.
2. Öffnen Sie die Detailseite des Rule Service, dessen Eigenschaften/Einstellungen Sie sehen wollen, indem Sie im Abschnitt "Rule Services" auf den zugehörigen Link in der Versionsspalte klicken.

Auf der Detailseite werden im Abschnitt "Übersicht" allgemeine Eigenschaften des Rule Service angezeigt:



Neben den Eigenschaften werden auch (veränderbare) Einstellungen des Rule Service angezeigt (siehe Abschnitt 2.1.5, „Rule Service Einstellungen“):



Die Änderung einer Einstellung können Sie folgendermaßen vornehmen:

1. Führen Sie für die einzustellende Eigenschaft die entsprechende Eingabe/Auswahl aus:
   - *Name der aktiven Konfiguration* : Eingabe des Namens der zu verwendenden Konfiguration
   - *Statistiklevel* : Auswahl eines Statistiklevels aus der Dropdown-Liste
   - *Gültigkeitszeitraum* : Eingabe eines Start- und/oder Endzeitpunktes (siehe Abschnitt 3.7.1.1, „Eingabe von Datum und Uhrzeit“)
   - *Aktiv* : Selektieren/Deselektieren der CheckBox um den Rule Service zu aktivieren/deaktivieren

   Möchten Sie alle vorgenommenen Eingaben wieder auf die ursprünglich gespeicherten Werte zurücksetzen, dann klicken Sie auf die Schaltfläche

.
2. Bestätigen Sie die Änderungen, indem Sie auf die Schaltfläche

klicken.

Das Editieren der Einstellungen ist nur möglich, wenn der angemeldete Benutzer über die entsprechende Berechtigung verfügt. Siehe auch Abschnitt 5.1, „Verfügbare Rollen und Berechtigungen“.

Weiterführende Konzepte.

- Abschnitt 2.1.5, „Rule Service Einstellungen“

3.2.8. Verwaltung der Metadaten eines Rule Service

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services aus.
2. Öffnen Sie die Detailseite des Rule Service, dessen Metadaten Sie verwalten wollen, indem Sie im Abschnitt "Rule Services" auf den zugehörigen Link in der Versionsspalte klicken. Links auf der Detailseite befindet sich die Aufgabenleiste für einen Rule Service:

3. Wählen Sie in der Aufgabenleiste den Eintrag Metadaten aus, worauf die Metadatenseite geöffnet wird.

Es gibt *modellbezogene Metadaten* eines Rule Service, die im Regelmodell gespeichert sind und nach der Bereitstellung des Rule Service nicht mehr verändert/gelöscht werden können. Name und Wert dieser Metadaten werden auf der Metadatenseite des Rule Service im Abschnitt "Modellbezogene Metadaten" angezeigt:

Modellbezogene Metadaten

Name	Wert
author	Erika Mustermann
entry rule	Preisberechnung

Zudem gibt es *zusätzliche Metadaten*, die Sie über die Webkonsole einem Rule Service nach dessen Bereitstellung zuordnen können. Diese Metadaten sind editierbar und werden im Abschnitt "Zusätzliche Metadaten" der Metadatenseite angezeigt:

Zusätzliche Metadaten

Metadaten hinzufügen

	Name	Wert
✖	priority	hoch

Das Editieren ist nur verfügbar, wenn der angemeldete Benutzer über die entsprechende Berechtigung verfügt. Siehe auch Abschnitt 5.1, „Verfügbare Rollen und Berechtigungen“.

3.2.8.1. Hinzufügen von Metadaten

Sie haben die Möglichkeit, einem Rule Service weitere Metadaten hinzuzufügen:

1. Klicken Sie im Abschnitt "Zusätzliche Metadaten" auf die Schaltfläche Metadaten hinzufügen. Eine leere Tabellenzeile wird hinzugefügt.
2. Geben Sie in der Spalte Name den neuen Metadatennamen ein.
3. Geben Sie in der Spalte Wert den Metadatenwert ein.
4. Klicken Sie auf die Schaltfläche Speichern. Die Metadatenliste wird um diesen Eintrag erweitert.

3.2.8.2. Löschen von Metadaten

Löschbare Metadaten bekommen im Abschnitt "Zusätzliche Metadaten" das Symbol ✖ zugeordnet.

1. Klicken Sie auf das Symbol ✖ des Metadatums, das Sie löschen möchten.
2. Es öffnet sich ein Dialog, in dem Sie bestätigen können, dass Sie die Aktion wirklich durchführen möchten.

   Klicken Sie zur Bestätigung auf die Schaltfläche Ja.

3.2.8.3. Editieren von Metadatennamen und –werten

An den zusätzlichen Metadaten lassen sich der Name und der Wert nachträglich editieren.

1. Klicken Sie auf den Metadatennamen bzw. -wert, um die Eingabe direkt in der grafischen Darstellung vorzunehmen.

2. Geben Sie den Metadatennamen bzw. -wert ein.

3. Klicken Sie auf die Schaltfläche

, um die Änderungen zu speichern.

Weiterführende Konzepte.

- Abschnitt 6.1, „Konzept Metadaten“

3.2.9. Anzeige der Ausführungen eines Rule Service

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services aus.

2. Öffnen Sie die Detailseite des Rule Service, dessen Ausführungen Sie sehen wollen, indem Sie im Abschnitt "Rule Services" auf den zugehörigen Link in der Versionsspalte klicken.

3. Wählen Sie in der Aufgabenleiste der Detailseite den Eintrag Ausführungen aus, worauf die Ausführungsseite geöffnet wird.

Die Ausführungsseite enthält den Abschnitt "Ausführungen", in dem alle Aufrufe dieses Rule Service angezeigt werden. Zudem gibt es den Abschnitt "Filter Ausführungen", in dem Sie die Anzeige der Ausführungen weiter einschränken können. Eine genauere Beschreibung finden Sie in Abschnitt 3.3.1, „Anzeige der Ausführungen von Rule Services“.

3.2.10. Anzeige der erforderlichen Bibliotheken eines Rule Service

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Rule Services aus.

2. Öffnen Sie die Detailseite des Rule Service, dessen erforderliche Bibliotheken Sie anzeigen möchten, indem Sie im Abschnitt "Rule Services" auf den zugehörigen Link in der Versionsspalte klicken.

3. Wählen Sie in der Aufgabenleiste der Detailseite den Eintrag Erforderliche Bibliotheken aus, worauf die Seite "*Erforderliche Bibliotheken*" geöffnet wird.

Die Seite "*Erforderliche Bibliotheken*" enthält den Abschnitt "Bibliotheken", in dem alle Bibliotheken, die dieser Rule Service benötigt, angezeigt werden. Eine genauere Beschreibung finden Sie in Abschnitt 3.4.1, „Anzeige bereitgestellter Bibliotheken“.

3.3. Verwaltung der Ausführungen von Rule Services

In der Sicht *Ausführungen* des Visual Rules Execution Servers können Sie Ausführungen von Rule Services verwalten.

Wählen Sie in der Menüleiste der Webkonsole den Eintrag Ausführungen, um diese Sicht zu öffnen.

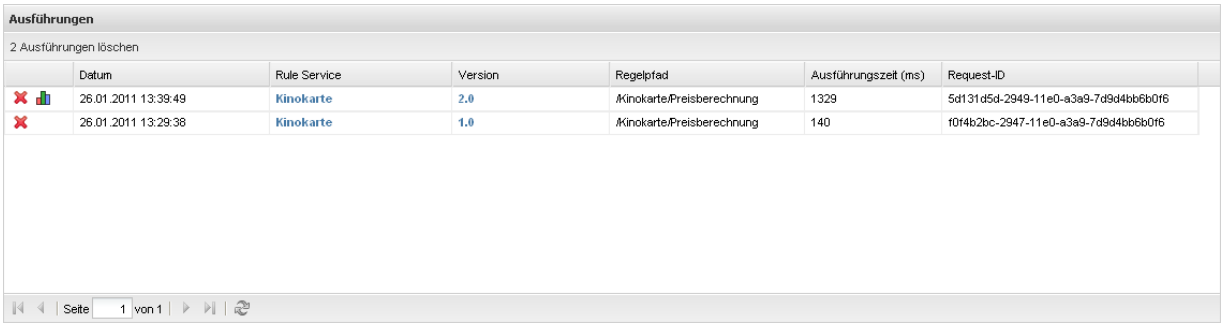
Hier haben Sie folgende Möglichkeiten:

- Abschnitt 3.3.1, „Anzeige der Ausführungen von Rule Services“
- Abschnitt 3.3.2, „Filterung angezeigter Ausführungen“
- Abschnitt 3.3.3, „Löschen von Ausführungen von Rule Services“
- Abschnitt 3.3.4, „Herunterladen einer Statistik zur Ausführung eines Rule Service“

3.3.1. Anzeige der Ausführungen von Rule Services

Wählen Sie in der Menüleiste der Webkonsole den Eintrag Ausführungen aus, um die Sicht *Ausführungen* zu öffnen.

Die geöffnete Übersichtsseite enthält den Abschnitt "Ausführungen", in dem die Aufrufe aller auf dem Visual Rules Execution Server bereitgestellten Rule Services angezeigt werden:



Standardmäßig werden folgende Eigenschaften der Ausführungen angezeigt:

- Ausführungszeitpunkt
- Name des aufgerufenen Rule Service
- Version des Rule Service
- Regelpfad
- Ausführungszeit
- Request-ID

Sie können das Layout der angezeigten Eigenschaften beeinflussen, indem Sie beispielsweise

- bestimmte Eigenschaften ein- bzw. ausblenden. Neben den standardmäßig angezeigten Eigenschaften gibt es noch weitere Eigenschaften wie *ID*, *Artefakt-ID* und *Gruppen-ID*, die eingeblendet werden können. (siehe Abschnitt 3.6.1, „Ein- und Ausblenden von Spalten“)
- zugehörige Spalten verschieben (siehe Abschnitt 3.6.2, „Verschieben einer Spalte“)
- das Sortierkriterium und die Sortierreihenfolge ändern (siehe Abschnitt 3.6.3, „Verändern des Sortierkriteriums und der Sortierreihenfolge“)

Falls bei der Ausführung eines Rule Service eine Statistik erstellt wurde, wird in der ersten Tabellenspalte das Symbol angezeigt.

Sind Sie nur an den Aufrufen/Ausführungen *eines* Rule Service interessiert, dann können Sie sich diese Ansicht über die Detailseite des Rule Service anzeigen lassen (siehe Abschnitt 3.2.9, „Anzeige der Ausführungen eines Rule Service“).

Weiterführende Aufgaben.

- Abschnitt 4.2.1, „Aufrufen eines Rule Service“
- Abschnitt 3.3.4, „Herunterladen einer Statistik zur Ausführung eines Rule Service“
- Abschnitt 3.3.3, „Löschen von Ausführungen von Rule Services“
- Abschnitt 3.6, „Konfiguration der Anzeige von Tabelleninhalten“

3.3.2. Filterung angezeigter Ausführungen

Auf der Übersichtsseite der Sicht *Ausführungen* befindet sich der Abschnitt "Filter Ausführungen", in dem Sie Filterkriterien spezifizieren können, um nur Ausführungen, die von Interesse sind, anzuzeigen:

In diesem Abschnitt können Sie die Request-ID der Ausführung als Filterkriterium angeben.

Sie haben die Möglichkeit, weitere Filterkriterien für die Anzeige der Ausführungen zu spezifizieren:

- Name des aufgerufenen Rule Service (in der Ausführungsseite eines Rule Services nicht verfügbar)
- Version des aufgerufenen Rule Service (in der Ausführungsseite eines Rule Services nicht verfügbar)
- Regelpfad
- Ausführungszeitraum

Dazu steht ein erweiterter Eingabedialog zur Verfügung:

Mehr Informationen zur Eingabe der Filterkriterien und der Anwendung des Filters finden Sie in Abschnitt 3.7, „Filterung angezeigter Objekte“.

3.3.3. Löschen von Ausführungen von Rule Services

Sie haben die Möglichkeit, eine bestimmte Ausführung oder alle im Abschnitt "Ausführungen" angezeigten Ausführungen zu löschen. Möchten Sie beispielsweise alle Ausführungen, die vor einem bestimmten Zeitpunkt liegen, löschen, dann können Sie sich durch Filterung der Ausführungen (Filterkriterium *Ausführungszeitraum*) die entsprechenden Ausführungen anzeigen lassen und darauf die Löschoperation ausführen.

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Ausführungen aus.
2. Wenn Sie eine einzelne Ausführung löschen möchten, dann klicken Sie im Abschnitt "Ausführungen" auf das Symbol der Ausführung, die Sie löschen möchten

   oder

   wenn Sie alle aktuell im Abschnitt "Ausführungen" angezeigten Ausführungen löschen möchten, dann klicken Sie dort auf die Schaltfläche , wobei n die Anzahl der betroffenen Ausführungen darstellt. Sofern nur eine Ausführung betroffen ist, heißt die Schaltfläche .
3. Es öffnet sich ein Dialog, in dem Sie bestätigen können, dass Sie die Aktion wirklich durchführen möchten. Klicken Sie zur Bestätigung auf die Schaltfläche .

Das Löschen ist nur möglich, wenn der angemeldete Benutzer über die entsprechende Berechtigung verfügt. Siehe auch Abschnitt 5.1, „Verfügbare Rollen und Berechtigungen“.

Weiterführende Aufgaben.

- Abschnitt 3.3.1, „Anzeige der Ausführungen von Rule Services“

3.3.4. Herunterladen einer Statistik zur Ausführung eines Rule Service

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Ausführungen aus.

2. Klicken Sie im Abschnitt "Ausführungen" bei der Ausführung, deren Statistik Sie herunterladen wollen, auf das Symbol . Das Symbol wird in der ersten Tabellenspalte nur angezeigt, wenn bei der Ausführung eine Statistik erstellt wurde.

3. Es öffnet sich ein Dialog zum Herunterladen der *vrstatistic*-Datei. Er ermöglicht Ihnen das Speichern der Datei. Das Aussehen des Dialoges ist abhängig von Ihrem Web Browser.

   Die heruntergeladene Statistik kann vom Visual Rules Modeler geladen und angezeigt werden (siehe Kapitel "Anzeigen von extern generierten Statistiken" im *Regel Modellierung Handbuch*).

Weiterführende Aufgaben.

- Abschnitt 3.3.1, „Anzeige der Ausführungen von Rule Services“

3.4. Verwaltung bereitgestellter Bibliotheken

In der Sicht *Bibliotheken* des Visual Rules Execution Servers können Sie bereitgestellte Bibliotheken verwalten.

Wählen Sie in der Menüleiste der Webkonsole den Eintrag Bibliotheken, um diese Sicht zu öffnen.

Hier haben Sie folgende Möglichkeiten:

- Abschnitt 3.4.1, „Anzeige bereitgestellter Bibliotheken“
- Abschnitt 3.4.2, „Filterung angezeigter Bibliotheken“
- Abschnitt 3.4.3, „Anzeige von Eigenschaften und Verwendung einer Bibliothek“
- Abschnitt 3.4.4, „Löschen einer bereitgestellten Bibliothek“

3.4.1. Anzeige bereitgestellter Bibliotheken

Wählen Sie in der Menüleiste der Webkonsole den Eintrag Bibliotheken aus, um die Sicht *Bibliotheken* zu öffnen.

Die geöffnete Übersichtsseite enthält den Abschnitt "Bibliotheken", in dem alle auf dem Visual Rules Execution Server bereitgestellten Bibliotheken angezeigt werden:

Bibliotheken

Artefakt-ID	Version	Gruppen-ID
activation	1.1	javax.activation
commons-collections	3.2.1	commons-collections
commons-lang	2.5	commons-lang
commons-logging	1.0.3	commons-logging
mail	1.4.2	javax.mail
stax	1.2.0	stax
stax-api	1.0	stax
visualrules-runtime	5.0.0-SNAPSHOT	de.visualrules
visualrules-runtime	5.0.0	de.visualrules
visualrules-runtime	5.1.0-SNAPSHOT	de.visualrules
xmlbeans	2.4.0	org.apache.xmlbeans

Seite 1 von 1

Standardmäßig werden folgende Eigenschaften der Bibliotheken angezeigt:

- Artefakt-ID
- Version
- Gruppen-ID

Sie können das Layout der angezeigten Eigenschaften beeinflussen, indem Sie beispielsweise

- bestimmte Eigenschaften ein- bzw. ausblenden. Neben den standardmäßig angezeigten Eigenschaften gibt es noch weitere Eigenschaften wie *Bereitgestellt von* und *Bereitgestellt am*, die eingeblendet werden können. (siehe Abschnitt 3.6.1, „Ein- und Ausblenden von Spalten“)
- zugehörige Spalten verschieben (siehe Abschnitt 3.6.2, „Verschieben einer Spalte“)
- das Sortierkriterium und die Sortierreihenfolge ändern (siehe Abschnitt 3.6.3, „Verändern des Sortierkriteriums und der Sortierreihenfolge“)

Sind Sie nur an den Bibliotheken interessiert, die *ein bestimmter* Rule Service benötigt, dann können Sie sich diese Ansicht über die Detailseite des Rule Service anzeigen lassen (siehe Abschnitt 3.2.10, „Anzeige der erforderlichen Bibliotheken eines Rule Service“).

Weiterführende Aufgaben.

- Abschnitt 3.4.4, „Löschen einer bereitgestellten Bibliothek“
- Abschnitt 3.6, „Konfiguration der Anzeige von Tabelleninhalten“

3.4.2. Filterung angezeigter Bibliotheken

Auf der Übersichtsseite der Sicht *Bibliotheken* befindet sich der Abschnitt "Filter Bibliotheken", in dem Sie Filterkriterien spezifizieren können, um nur Bibliotheken, die von Interesse sind, anzuzeigen:

In diesem Abschnitt können Sie die Artefakt-ID der Bibliothek als Filterkriterium angeben.

Sie haben die Möglichkeit, weitere Filterkriterien für die Anzeige der Bibliotheken zu spezifizieren:

- Gruppen-ID
- Version
- Bereitstellungszeitraum

Dazu steht ein erweiterter Eingabedialog zur Verfügung:

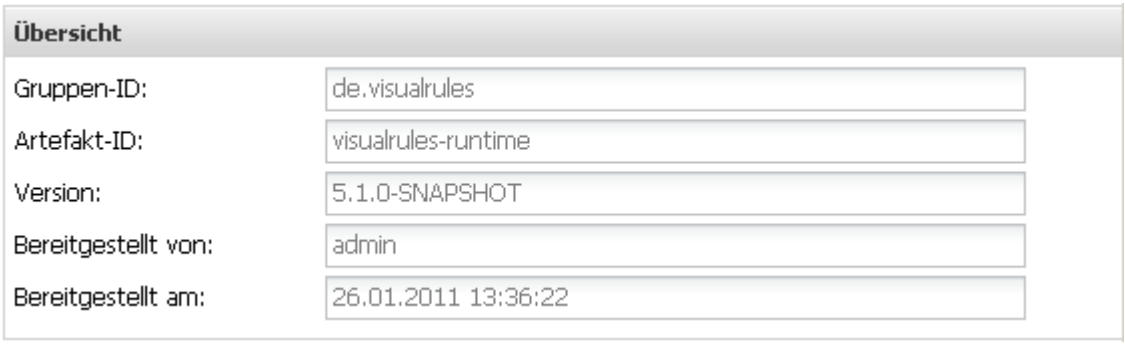
Mehr Informationen zur Eingabe der Filterkriterien und der Anwendung des Filters finden Sie in Abschnitt 3.7, „Filterung angezeigter Objekte“.

3.4.3. Anzeige von Eigenschaften und Verwendung einer Bibliothek

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Bibliotheken aus.

2. Öffnen Sie die Detailseite der Bibliothek, deren Eigenschaften/Verwendung Sie sehen möchten, indem Sie im Abschnitt "Bibliotheken" auf den zugehörigen Link in der Spalte *Artefakt-ID* klicken.

Auf der Detailseite werden im Abschnitt "Übersicht" allgemeine Eigenschaften der Bibliothek angezeigt:



Neben den Eigenschaften werden auf dieser Seite im Abschnitt "Benötigt von Rule Services" alle Rule Services angezeigt, die diese Bibliothek verwenden:

Eine genauere Beschreibung der Tabelle finden Sie in Abschnitt 3.2.1, „Anzeige bereitgestellter Rule Services“.

3.4.4. Löschen einer bereitgestellten Bibliothek

1. Wählen Sie in der Menüleiste der Webkonsole den Eintrag Bibliotheken aus.

2. Öffnen Sie die Detailseite der Bibliothek, die Sie löschen wollen, indem Sie im Abschnitt "Bibliotheken" auf den zugehörigen Link in der Spalte *Artefakt-ID* klicken.

3. Klicken Sie auf die Schaltfläche

.

Diese Schaltfläche ist nur dann aktiviert, wenn diese Bibliothek von keinem anderen Rule Service mehr benötigt wird und wenn Sie ausreichende Berechtigungen haben. Der Standardbenutzer `admin` hat diese Erlaubnis. Siehe auch Abschnitt 5.1, „Verfügbare Rollen und Berechtigungen“.

4. Es öffnet sich ein Dialog, in dem Sie bestätigen können, dass Sie die Aktion wirklich durchführen möchten.

   Klicken Sie zur Bestätigung auf die Schaltfläche

.

3.5. Wartung des Execution Servers

3.5.1. Anzeige von Lizenzinformationen

Wählen Sie in der Menüleiste der Webkonsole den Eintrag Wartung aus.

Die geöffnete Lizenzseite enthält den Abschnitt "Lizenzinformationen", in dem Informationen über alle Lizenzen angezeigt werden, die für den Visual Rules Execution Server relevant sind:

Lizenzinformationen

	Datei	Lizenznehmer	Typ	Version	Wartung endet am	Gültig
1	license_exec_server_1.txt	-	Evaluation	4	19.01.2010	18.02.2009
2	license_exec_server_2.txt	Exec Server	Full	4	23.02.2010	25.03.2010

Es werden folgende Lizenzeigenschaften angezeigt:

- Name der Lizenzdatei
- Lizenznehmer
- Lizenztyp
- Version
- Wartungsende
- Gültigkeit

Weiterführende Konzepte.

- Abschnitt 1.4, „Installation der Lizenz“

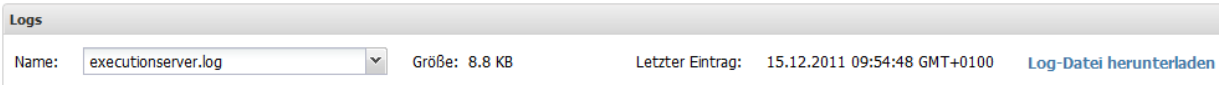
3.5.2. Konfiguration und Anzeige von Nachrichten der Laufzeitprotokollierung

Wählen Sie in der Menüleiste der Webkonsole den Eintrag Wartung aus. Selektieren Sie danach den Punkt "Logging".

Die geöffnete Seite besteht aus zwei Teilen. Der obere, mit "Detaillierungsgrad" beschriebene Teil, dient zur Konfiguration der Laufzeitprotokollierung des Execution Server. Dazu dient ein Schieberegeler:

Der Schieberegeler kontrolliert die Einstellung des Detaillierungsgrades der Laufzeitprotokollierung. Je höher der Grad, desto mehr Nachrichten werden produziert. Zu beachten ist, dass ein hoher Detaillierungsgrad einen negativen Einfluss auf die Leistungsfähigkeit des Systems hat. Neben dem Schieberegler wird eine genaue Beschreibung für den momentan eingestellten Detaillierungsgrad angezeigt.

Der untere Teil der Seite, ist mit "Logs" beschrieben und zeigt hauptsächlich Protokolldateien (Log) an:



Die Dropdown-Liste wählt die Protokolldatei aus, deren Einträge angezeigt werden sollen. Eine Datei kann auch leer sein. Informationen zu existierenden Protokolldateien ist in Abschnitt 1.7, „Laufzeitprotokollierung“ beschrieben. Mit dem "Log-Datei herunterladen" beschrifteten Link, kann die ausgewählte Protokolldatei heruntergeladen werden.

3.6. Konfiguration der Anzeige von Tabelleninhalten

Die tabellarische Anzeige von Informationen in der Webkonsole kann oftmals auf folgende Art und Weise verändert werden:

- Abschnitt 3.6.1, „Ein- und Ausblenden von Spalten“
- Abschnitt 3.6.2, „Verschieben einer Spalte“
- Abschnitt 3.6.3, „Verändern des Sortierkriteriums und der Sortierreihenfolge“

3.6.1. Ein- und Ausblenden von Spalten

In vielen Tabellen ist es möglich, die Anzeige bestimmter Spalten zu aktivieren/deaktivieren:

1. Bewegen Sie die Maus über die Überschrift einer beliebigen Spalte der Tabelle. Klicken Sie auf , um das Kontextmenü zu öffnen.
2. Gehen Sie mit der Maus auf den Menüpunkt Spalten, worauf sich ein weiteres Kontextmenü öffnet, in dem ausblendbare Spalten aufgeführt werden.
3. Aktivieren/Deaktivieren Sie die Checkbox der Spalten, die Sie ein-/ausblenden möchten.

3.6.2. Verschieben einer Spalte

In vielen Tabellen haben Sie die Möglichkeit, die Reihenfolge der Spalten zu verändern:

1. Eine Spalte kann per Drag&Drop an eine andere Position verschoben werden. Klicken Sie auf die Überschrift der Spalte, die Sie verschieben möchten, und halten Sie die linke Maustaste gedrückt.
2. Bewegen Sie die Maus nach links bzw. rechts. Während der Mausbewegung erscheinen Symbole, die signalisieren, ob es sich bei der aktuellen Position um eine mögliche oder unerlaubte Position handelt. Zusätzlich zeigen Pfeile die genaue (mögliche) Position an.
3. Haben Sie die gewünschte Spaltenposition erreicht, dann lassen Sie die Maustaste los. Die Spalte wird nun dorthin verschoben.

3.6.3. Verändern des Sortierkriteriums und der Sortierreihenfolge

1. Klicken Sie auf die Spaltenüberschrift, um die zugehörige Eigenschaft als Sortierkriterium festzulegen. Das Symbol bedeutet, dass *aufsteigend* sortiert wird.
2. Falls Sie eine absteigende Sortierreihenfolge möchten, dann klicken Sie erneut auf die Spaltenüberschrift. Das Symbol bedeutet, dass *absteigend* sortiert wird.

Alternativ können Sie die Sortierreihenfolge auch über das Kontextmenü der Spalte festlegen:

1. Bewegen Sie die Maus über die Spaltenüberschrift und klicken Sie auf , um das Kontextmenü zu öffnen.
2. Selektieren Sie den Menüpunkt Aufsteigend sortieren bzw. Absteigend sortieren.

3.7. Filterung angezeigter Objekte

Durch die Anwendung eines Filters kann die Zahl der angezeigten Objekte in einer Tabelle reduziert und der Fokus auf bestimmte Objekte gelegt werden.

3.7.1. Eingabe von Filterkriterien

Filterkriterien werden in einem eigenen Abschnitt der Benutzeroberfläche spezifiziert. Der Filterabschnitt befindet sich meist vor der Tabelle, auf deren Objekte sich die Filterung beziehen soll. Initial ist die Eingabe *eines* (wichtigen) Filterkriteriums möglich.

Um weitere Filterkriterien eingeben zu können, können Sie in einen erweiterten Eingabedialog mit zusätzlichen Eingabefeldern wechseln. Dazu gibt es im Filterabschnitt die Schaltfläche

.

Den erweiterten Dialog können Sie wieder verlassen, indem Sie dort auf die Schaltfläche

klicken.

Sie können die Eingabe aller Filterkriterien rückgängig machen, indem Sie auf die Schaltfläche

klicken. Die Filtereingabefelder sind danach leer. Zudem wird die Anzeige der zugehörigen Tabelle aktualisiert.

3.7.1.1. Eingabe von Datum und Uhrzeit

Die Eingabefelder zur Spezifikation eines Datums und einer Uhrzeit haben folgendes Aussehen:

Zur Eingabe eines Datums klicken Sie auf das Symbol , sodass sich ein Kalender öffnet:

Hier können Sie per Mausklick ein Datum auswählen, das dann automatisch in das Datumsfeld übernommen wird.

Der Kalender bietet auch die Möglichkeit, den Monat und das Jahr zu wechseln. Klicken Sie dazu auf die Monatsüberschrift im Kalender, sodass sich folgende Ansicht zur Auswahl öffnet:

In dieser Ansicht können Sie mit der Maus einen Monat in einem bestimmten Jahr auswählen. Bestätigen Sie Ihre Wahl anschließend mit OK.

Zur Eingabe einer Uhrzeit klicken Sie auf das Symbol neben dem zweiten Eingabefeld. Es öffnet sich eine Liste, aus der Sie einen Zeitpunkt (Stunden und Minuten) auswählen können.

3.7.2. Anwenden eines Filters

Soll die Filterung der Objekte einer Tabelle ausgeführt werden, klicken Sie im Filterabschnitt auf die Schaltfläche

. Anschließend wird die Anzeige der zugehörigen Tabelle aktualisiert und es werden nur solche Objekte angezeigt, deren Eigenschaften mit den spezifizierten Filterkriterien übereinstimmen.

Kapitel 4. Aufrufen von Regeln im Execution Server

4.1. Konzepte

4.1.1. Format der Rule Service Anfrage

Das äußerste Element einer Anfrage an eine Regel wird `VRRequest` genannt. Der Namespace dieses Elements reflektiert die Position der Regel im Regelmodell und in der Regelpakethierarchie. Diese Information wird vom Execution Server verwendet, um die Anfrage an das gewünschte Regelmodell und an die darin gewünschte Regel weiterzuleiten.

Der vollständige Namespace für den URI des `VRRequest` besteht aus dem Präfix, der im Regelmodell konfiguriert werden kann, gefolgt vom Segment `/vrpath`, gefolgt von dem Pfad der Regel, die aufgerufen werden soll. Der Pfad besteht aus dem Namen des Regelmodells, desweiteren den Namen der Regelpakete und schliesslich des Namens der Regel selbst.

Der Namespace URI Präfix ist standardmäßig `http://www.visual-rules.com` und wird in den folgenden Beispielen verwendet.

Die nachfolgende Beschreibung bezieht sich auf die aktuelle Version. Bei der Verwendung von migrierten Rule Services ist die Web Service Schnittstelle immer noch richtig, enthält aber nicht alle hier beschriebenen Elemente.

Das `VRRequest` Element kann ein optionales `target` Element beinhalten, mit dem eine spezifische Version des aufzurufenden Regelmodells angeben werden kann. Wenn das `target` Element nicht vorhanden ist, ist es das Standardverhalten des Execution Server, die neueste Version des Regelmodells aufzurufen.

Als nächstes kommt das `configuration` Element, dass Einstellungen zur Ausführung des Rule Services spezifiziert und ebenfalls optional ist.

Alle Elemente haben eine definierte Reihenfolge. Ist das `target` Element spezifiziert, so muss es das erste Element im `VRRequest` sein und das `configuration` Element käme als nächstes, wenn es spezifiziert ist.

Schließlich werden die Eingabedaten der Regel in einem Element `input` spezifiziert. Die Elemente in diesem Abschnitt haben den gleichen Namen wie die Eingaben einer Regel. Der Wert jedes Datenelements wird innerhalb dieser Elemente spezifiziert. Der Abschnitt 4.1.4, „XML Repräsentation von Datentypen“ beschreibt die XML Repräsentation für diese Werte, abhängig von den Datentypen (einschließlich Strukturen, Aufzählungen, Listen, Mengen und Maps).

```
<pricing:VRRequest xmlns:vr="http://www.visual-rules.com" xmlns:pricing="http://www.visual-rules.com/vrpath/M
   <!-- Falls weggelassen, wird Visual Rules Execution Server die
        "letzte" Version verwenden -->
   <vr:target>
      <version>1.0.1</version>
   </vr:target>
   <!-- Spezifiziert optionale Einstellungen für die Ausführung eines Rule Service. -->
   <vr:configuration>
      <!-- Eine requestId kann spezifiziert werden, um die Statistik auf dem Server
           zu identifizieren. Falls weggelassen, wird eine eindeutige ID generiert. -->
      <requestId>uniqueRequestId</requestId>

      <!-- Der Name der aktiven Konfiguration (active configuration name) der während der
           Ausführung verwendet werden soll.
           Wird dieser weggelassen, wird der Standard verwendet. -->
      <activeConfigurationName>test</activeConfigurationName>

      <!-- Für ältere Aufrufer, kann anstatt activeConfigurationName auch binding noch
           verwendet werden. -->
      <!-- Ebenso sollte dies für Rule Services verwendet werden, die mit einer
           Version vor 5.2 gebaut wurde. -->
      <binding>test</binding>

      <!-- Der verwendete Statistik Level. Mögliche Angaben sind: high, medium und low -->
      <!-- Falls weggelassen, werden keine Statistiken aufgezeichnet und sind
           auch nicht herunterladbar. -->
      <sessionStatistics>
          <level>medium</level>
      </sessionStatistics>
   </vr:configuration>
   <input>
      <auditorium_no>1</auditorium_no>
      <seat_no>199</seat_no>
      <show_date>2008-08-22</show_date>
      <coupon>true</coupon>
      <student>true</student>
      <bonus_card>GOLD</bonus_card>
   </input>
</pricing:VRRequest>
```

Beispiel 4.1. VRRequest Format

Weiterführende Konzepte.

- Abschnitt 4.1.4, „XML Repräsentation von Datentypen“

4.1.2. Format der Rule Service Antwort

Das äußerste Element einer Antwort auf eine Regel Anfrage wird `VRResponse` genannt. Der Namespace ist der gleiche wie für die Anfrage.

Die `VRResponse` hat ein Element mit dem Namen `output`, dass die Werte für alle Ausgabedatenelemente einer Regel enthält. Im Normalfall enthält die Antwort auch Aktionen. Es ist zu beachten, dass dies konfiguriert werden kann, wie in Abschnitt 2.2.3, „Einstellen von Aktionen als Rückgabewert“ erläutert. Der Abschnitt 4.1.4, „XML Repräsentation von Datentypen“ beschreibt die XML Repräsentation für die Werte hier.

Schließlich gibt es ein Element `trace`, das die Informationen darüber zurückgibt, welche Regel in welcher Version von welchem Regelmodell gerade aufgerufen wurde. Dies dient dem Zweck der Nachverfolgbarkeit, so dass immer klar erkenntlich ist, welche Regeln ausgeführt wurden, beispielsweise wenn in der Anfrage die zu verwendende Version nicht festgelegt wurde.

Um die Statistiken nach der Ausführung auf dem Execution Server zu identifizieren, wird die angegebene oder generierte `requestId` zurückgeliefert. Mehr zu Statistiken ist in Abschnitt 3.3.1, „Anzeige der Ausführungen von Rule Services“ zu finden. Als letztes Element wird der Name der aktiven Konfiguration (`active configuration name`) ausgegeben, der entweder in der Anfrage oder als Einstellung des Rule Service spezifiziert wurde. Auch hier wird aus Kompatibilitätsgrüunden für ältere Rule Services das Element `binding` zurückgeliefert.

```
<pricing:VRResponse xmlns:vr="http://www.visual-rules.com" xmlns:pricing="http://www.visual-rules.com/vrpath/
   <output>
      <price>7</price>
   </output>
   <vr:trace>
      <ruleModel>Movie Ticket Pricing</ruleModel>
      <rulePath>/Movie Ticket Pricing/Pricing</rulePath>
      <version>1.0.1</version>
      <requestId>3919c5b8-5bfc-11de-b728-d56c9d794642</requestId>
      <!-- Optional -->
      <activeConfigurationName>test</activeConfigurationName>
      <!-- oder -->
      <binding>test</binding>
   </vr:trace>
</pricing:VRResponse>
```

Beispiel 4.2. VRResponse Format

Weiterführende Konzepte.

- Abschnitt 4.1.4, „XML Repräsentation von Datentypen“

4.1.3. Generische Rule Service Anfragen

Der gewöhnliche Weg um einen Rule Service aufzurufen, besteht darin eine Anfrage aus der spezifischen WSDL zu erzeugen und diese abzuschicken. Dieses Vorgehen bedeutet auch, dass sich der Aufrufer bereits für einen Rule Service und den damit verbundenen Regeln entschieden hat. Es gibt jedoch auch Fälle, in denen es gewünscht ist, gegen eine für alle Rule Services gleiche WSDL zu arbeiten und die Anfrage daraus zu erstellen. Dies setzt eine implizite Kenntniss der Daten, die zu übergeben sind, voraus. Die zweite Anwendung der generischen Anfrage ist das Weiterleiten einer Anfrage zu einem Rule Service anhand von Metadaten.

Es ist ebenso möglich eine spezifische WSDL anhand von Metadaten abzuholen. Mehr dazu in Abschnitt 6.2.1.2, „Eine WSDL durch Angabe von Metadata abholen“

4.1.3.1. Generisches VRRequest Format

Der generische VRRequest ist in einer speziellen WSDL Datei spezifiziert, die auf dem Execution Server unter `http://<server>:<port>/<context-name>/services/visualrules_generic.wsdl` verfügbar ist. In einer Standard Installation auf einem lokalen Rechner könnte dies beispielsweise so aussehen (wobei x und y der Versionsnummer der WAR Datei des Execution Servers entsprechen):

```
http://localhost:8080/executionserver-5.x.y/services/visualrules_generic.wsdl
```

Es gibt einige Ähnlichkeiten zwischen dem spezifischen und dem generischen Anfrage Format. Das `VRRequest` Element ist ebenfalls das äußerste Element in der generischen Anfrage allerdings mit einem festen Namespace. Weil die Information des Regel Modells und der aufzurufenden Regel nicht mehr im Namespace vorhanden sind, müssen diese nun mit `ruleModel` und `rulePath` im `target` Element spezifiziert werden. Deswegen ist `target` auch nicht mehr optional. Es kann optional eine `version` angegeben werden, was die gleiche Bedeutung wie im spezifischen `VRRequest` hat.

Als Alternative zur `version` kann ein Element `effectiveDate` angegeben werden, welches die Anfrage anhand der Metadaten zu einem Rule Service leitet. In Abschnitt 6.2.1.1, „Standard-Metadata Mapper“ ist die Funktionsweise genauer erklärt. Zu beachten ist, dass die Elemente, welche unter `target` eingetragen werden ebenfalls angepasst werden können, wenn ein benutzerdefinierter `Matadata Mapper` eingesetzt wird, wie das in Abschnitt 6.2.1.3, „Benutzerdefinierter Metadata Mapper“ beschrieben ist.

Das nächste Element ist `configuration`, dass genau dasselbe ist, wie in Abschnitt 4.1.1, „Format der Rule Service Anfrage“ beschrieben. Als nächstes kommt das `input` Element, welches beliebige Elemente akzeptiert. Das ist der Nachteil des generischen Ansatzes und dies ist zugleich der größte Unterschied zur spezifischen Anfrage, die genau beschriebt, welche Elemente und Typen enthalten sind.

```
<gen:VRRequest xmlns:vr="http://www.visual-rules.com" xmlns:gen="http://www.visual-rules.com/generic" >
   <target>
      <ruleModel>Movie Ticket Pricing</ruleModel>
      <rulePath>/Movie Ticket Pricing/Pricing</rulePath>
      <!--Eines der beiden Elemente kann angegeben werden:-->
      <effectiveDate>2009-10-03</effectiveDate>
      <!-- oder -->
      <version>1.0.1</version>
   </target>
   <!-- Optional -->
   <vr:configuration>
      <!-- Die gleichen Elemente wie in der spezifischen Anfrage -->
   </vr:configuration>
   <input>
      <!-- Diese Elemente werden nicht vom XML Schema definiert. Die hier verwendeten
           Daten und Typen müssen vom Regelinterface akzeptiert werden. -->
      <auditorium_no>1</auditorium_no>
      <seat_no>199</seat_no>
      <show_date>2008-08-22</show_date>
      <coupon>true</coupon>
      <student>true</student>
      <bonus_card>GOLD</bonus_card>
   </input>
</gen:VRRequest>
```

Die Antwort einer generischen Anfrage unterscheidet sich nicht von der einer spezifischen Anfrage.

Related Concepts.

- Abschnitt 4.1.4, „XML Repräsentation von Datentypen“

Beispiel 4.3. Generic VRRequest format

4.1.4. XML Repräsentation von Datentypen

4.1.4.1. Einfache Typen

Einfache Typen werden auf folgende XML Schema Typen abgebildet:

Visual Rules Datentyp	XML Schema Typ
`String`	`xsd:string`
`Integer`	`xsd:integer`
`Float`	`xsd:decimal`
`Boolean`	`xsd:boolean`
`Date`	`xsd:date`
`Time`	`xsd:time`
`Timestamp`	`xsd:dateTime`

`String` Werte werden einfach als Text abgebildet (ohne Anführungszeichen). Zum Beispiel wird in diesem XML Fragment für ein Datenelement `name` ein String Wert `"Peter"` angegeben.

```
<name>Peter</name>
```

`Integer` Werte werden als Ganzzahlen abgebildet. Zum Beispiel wird in diesem XML Fragment 199 als der ganzzahlige Wert für ein Datenelement `seat_no` angegeben.

```
<seat_no>199</seat_no>
```

`Float` Werte werden als Gleitkommazahlen dargestellt, unter Verwendung eines Punkt als Dezimaltrennzeichen. Zum Beispiel wird in diesem XML Fragment für verschiedene Datenelemente jeweils eine Gleitkommazahl

als Wert angegeben. Bitte lesen Sie die XML Schema Dokumentation für detaillierte Beschreibungen von `xsd:decimal`.

```
<price>7.45</price>
<rate>-.4453</rate>
```

`Boolean` Werte werden durch das Wort `true` oder `false` abgebildet. Der XML Typ ist `xsd:boolean`.

```
<student>true</student>
<coupon>false</coupon>
```

`Date` Werte entsprechen dem Format: `YYYY-MM-DD`. Dabei stehen die ersten vier Zeichen für das Jahr, dann zwei Zeichen für den Monat und schließlich zwei Zeichen für den Tag, wobei alle durch Bindestriche (-) getrennt sind. Bitte lesen Sie die XML Schema Dokumentation für eine detaillierte Beschreibung von `xsd:date`.

```
<show_date>2008-08-22</show_date>
```

`Time` Werte entsprechen dem Format: `hh:mm:ss`. Die ersten zwei Zeichen stehen für die Stunde, dann Minuten und Sekunden, jeweils getrennt durch Doppelpunkte (:). Kein Bestandteil kann weggelassen werden. Bitte lesen Sie die XML Schema Dokumentation für eine detaillierte Beschreibung von `xsd:time`.

```
<alarm>15:47:23</alarm>
```

`Timestamp` Werte werden entsprechend dem XML Schema Typ `xsd:dateTime` repräsentiert. Das Format ist `YYYY-MM-DDThh:mm:ss`, was das Datum und die Zeit getrennt durch den Buchstaben `T` darstellt. Kein Bestandteil darf weggelassen werden. Bitte lesen Sie die XML Schema Dokumentation für eine detaillierte Beschreibung für das Format von `xsd:dateTime`.

```
<startTimestamp>2009-05-30T09:30:10</startTimestamp>
```

Sofern Werte ausdrücklich als leer angegeben werden sollen (in Java durch `null` repräsentiert), kann dies durch Angabe das `xsi:nil` Attributes erreicht werden. Dazu ist es erforderlich, in der Anfrage auch den Namespace `xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"` zu importieren.

```
<myDate xsi:nil="true" />
```

Beachten Sie, dass die Angabe des `xsi:nil` Attributes nur für Datentypen funktioniert, die dies unterstützen. Diese Angabe ist beispielsweise wirkungslos, wenn es sich in Java um einen primitiven Datentyp wie `int` handelt, da diese ihrer Natur nach immer einen Wert haben.

4.1.4.2. Strukturen

Werte von Strukturen in einer Anfrage oder einer Antwort werden einfach durch Elemente von jedem Attribut repräsentiert. Zum Beispiel repräsentiert das folgende XML Fragment den Wert des Datenelements `customer` mit zwei Attributen `name` und `address`. Dabei hat `address` selbst ein Attribut `zip`.

```
<customer>
   <name>John Doe</name>
   <address>
      <zip>12345</zip>
   </address>
</customer>
```

4.1.4.3. Listen und Mengen

Die Werte einer Liste oder Menge in einer Anfrage oder Antwort wird durch eine Reihe von `element` Tags abgebildet.

Zum Beispiel sieht eine Liste (oder Menge) von Strings so aus:

```
<names>
   <element>John Doe</element>
   <element>Peter Pan</element>
   <element>Captain Hook</element>
</names>
```

Und dies ist eine Liste (oder Menge) von Kunden, wobei jeder Kunde die Attribute `name` und `address` hat:

```
<customers>
   <element>
      <name>John Doe</name>
      <address>
         <zip>12345</zip>
      </address>
   </element>
   <element>
      <name>Peter Pan</name>
      <address>
         <zip>99999</zip>
      </address>
   </element>
   <element>
      <name>Captain Hook</name>
      <address>
         <zip>99996</zip>
      </address>
   </element>
</customers>
```

4.1.4.4. Maps

Maps werden durch eine Reihe von `entry` Tags spezifiziert, jede davon hat zwei Elemente, `key` und `value` genannt. Zum Beispiel ist dies ein `String -> Customer` Mapping mit zwei Einträgen:

```
<customerMap>
   <entry>
      <key>8847-736-90</key>
      <value>
         <name>John Doe</name>
         <address>
            <zip>12345</zip>
         </address>
      </value>
   </entry>
   <entry>
      <key>2234-993-77</key>
      <value>
         <name>Peter Pan</name>
         <address>
            <zip>99999</zip>
         </address>
      </value>
   </entry>
</customerMap>
```

4.1.4.5. Aufzählungen

Der Wert für eine Aufzählung wird einfach durch Angabe des gewünschten Literals definiert. Im folgende Beispiel wird für eine Aufzählung für das Datenelement `bonus_card` der Wert `GOLD` angegeben:

```
<bonus_card>GOLD</bonus_card>
```

4.1.5. Abbildung des Regelmodells auf WSDL

Wird ein Regelmodell als Rule Service exportiert, so wird die WSDL Datei mit folgender Prozedur erstellt:

- Die WSDL enthält einen Service, der `<rule-model>Service` genannt wird.
- Jede exportierte Regel wird durch eine Operation (in einem portType) repräsentiert, die mit dem Regelnamen endet.

Die WSDL Datei importiert einige XML Schema Dateien. Die XML Schema Dateien werden folgendermaßen erstellt:

- Es gibt eine XML Schema Datei für jede Ebene im Regelmodell, d.h. eine für das Regelmodell selbst, eine für jedes Regelpaket und eine für jede Regel (Ablaufregel oder Entscheidungstabelle), die für das Exportieren als Rule Service gekennzeichnet wurde.
- Jede XML Schema Datei beinhaltet die Definitionen der Datentypen, die auf dieser Ebene des Regelmodells vorhanden sind.
- Die Namespace URI jedes XML Schemas ist vom Regelpaketnamen abgeleitet, d.h. Sie werden im XML Schema Namespace die gleiche Hierarchie der Regelpakete und Regeln finden.
- Die XML Schema Dateien für exportierte Regeln enthalten auch die Definitionen des Nachrichtenformats für die Input und Output Datenelemente. Anders ausgedrückt wird damit die Schnittstelle der Regel, einschließlich der Namen und Typen aller Eingabe/Ausgabe Datenelemente, beschrieben.

Weiterführende Arbeitsschritte.

- Abschnitt 3.2.5, „Anzeige der WSDL-Datei eines Rule Service“

4.2. Arbeitsschritte

4.2.1. Aufrufen eines Rule Service

Um einen Rule Service im Execution Server aufzurufen, müssen Sie eine SOAP Anfrage an den HTTP Endpunkt schicken, der in der WSDL spezifiziert wurde. Standardmässig ist dieser Endpunkt der gleiche für alle Rule Services (was es einfacher für generische Clients macht).

Das Vorgehen, um einen Web Service aufzurufen, hängt stark von der Programmiersprache oder dem Tool ab, das Sie verwenden und kann hier nicht beschrieben werden. So wird in den folgenden Beschreibungen lediglich erklärt, was technisch und im SOAP Protokoll geschieht.

1. Im SOAP Nachrichtenteil spezifiziert der Web Service Client einen `VRRequest`, wie im Abschnitt 4.1.1, „Format der Rule Service Anfrage“ beschrieben. Beispielsweise kann dies so aussehen:

```
<pricing:VRRequest xmlns:pricing="http://www.visual-rules.com/vrpath/Movie%20Ticket%20Pricing/Pricing">
   <input>
      <auditorium_no>1</auditorium_no>
      <seat_no>199</seat_no>
      <show_date>2008-08-22</show_date>
      <coupon>true</coupon>
      <student>true</student>
      <bonus_card>GOLD</bonus_card>
   </input>
</pricing:VRRequest>
```

Nur der XML Namespace des `VRRequest` Elements wird vom Execution Server verwendet, um die Anfrage an das korrekte Regelmodell weiterzuleiten. Die URL des Service Endpunkts oder der `SOAPAction` HTTP Header sind nicht relevant.

2. Optional kann das `target` Element als das erste Element im `VRRequest` hinzugefügt werden, um genau zu spezifizieren, welche Version des Regelmodells aufgerufen werden soll. Wenn dies weggelassen wird, ruft der Execution Server die "neueste" Version auf.

3. Wenn der Regelaufruf erfolgreich ist, erhalten Sie eine `VRResponse` wie in Abschnitt 4.1.2, „Format der Rule Service Antwort“ beschrieben. Die Ergebnisse des Regelaufrufs können im `output` Element der `VRResponse` gefunden werden. Das zusätzliche Trace-Element beinhaltet Informationen über das aufgerufene Regelmodell, die Regel und die Version.

```
<pricing:VRResponse xmlns:vr="http://www.visual-rules.com" xmlns:pricing="http://www.visual-rules.com/vrp
   <output>
      <price>7</price>
   </output>
   <vr:trace>
      <ruleModel>Movie Ticket Pricing</ruleModel>
      <rulePath>/Movie Ticket Pricing/Pricing</rulePath>
      <version>1.0.1</version>
   </vr:trace>
</pricing:VRResponse>
```

Wenn irgendetwas fehlschlägt, erhalten Sie eine SOAP Fehlernachricht. Die Fehlernachricht enthält unter anderem einen Java Stack Trace für die Fehlerdiagnose.

Weiterführende Konzepte.

- Abschnitt 4.1.1, „Format der Rule Service Anfrage“
- Abschnitt 4.1.2, „Format der Rule Service Antwort“

Weiterführende Arbeitsschritte.

- Abschnitt 3.2.5, „Anzeige der WSDL-Datei eines Rule Service“

Kapitel 5. Rollen des Sicherheitskonzepts

5.1. Verfügbare Rollen und Berechtigungen

Ein Benutzer des Execution Servers hat mindestens die Rolle `ROLE_USER`. Um administrative Aufgaben wahrnehmen zu können, muss ein Benutzer zusätzlich über die Rolle `ROLE_ADMIN` verfügen.

Tabelle 5.1. Übersicht der Berechtigung von ROLE_ADMIN

Bereitstellung von Rule Services / Artefakten
Bereitstellung von Rule Services / Artefakten beenden
Hochladen von Visual Rules Archiv Dateien
Editieren von Einstellungen von Rule Services
Editieren von Metadaten von Rule Services
Löschen von Ausführungen von Rule Services

Die Konfiguration von Benutzern und ihren Rollen ist in Abschnitt 1.5.1, „Konfiguration der Authentifizierung und Berechtigungen“ beschrieben.

Kapitel 6. Arbeiten mit Metadaten

6.1. Konzept Metadaten

Metadaten werden durch einfache Schlüssel-Wert Paare abgebildet und können zu jedem Rule Service hinzugefügt werden. In erster Linie dienen sie als erweiterte Informationen für ein Regelmodell. Beispielsweise könnte dies der Name eines bestimmten Algorithmus sein, der für eine Berechnung eingesetzt wird, oder auch einfach nur der Autor der Regeln.

Wenn Sie in einem multinationalen Team arbeiten, sollten Sie passende Namen für Metadaten wählen, damit diese von jedem verstanden werden.

Mit Hilfe von Metadaten ist es möglich eine WSDL für einen Rule Service abzuholen und der Execution Server ist in der Lage damit generische Anfragen zu einem Rule Service zu leiten. Das hängt von der eingesetzten Implementierung der `Metadata Mapper` Komponente ab. Eine Standardimplementierung einer `Metadata Mapper` Komponente ist in Abschnitt 6.2.1.1, „Standard-Metadata Mapper“ beschrieben. Es ist auch möglich eine benutzerdefinierte Implementierung zu verwenden. Letzteres ist in Abschnitt 6.2.1.3, „Benutzerdefinierter Metadata Mapper“ beschrieben.

6.2. Metadaten definieren

Metadaten bestehen aus einer Menge eindeutiger Schlüssel und zugehöriger Werte. Es gibt zwei Wege, um Metadaten zu bearbeiten. Sie können nach der Bereitstellung in der Webkonsole eingegeben werden (was inAbschnitt 3.2.8, „Verwaltung der Metadaten eines Rule Service“ beschrieben ist) oder vor der Bereitstellung im Visual Rules Modeler. Dabei werden die Metadaten im Regelmodell gespeichert und bilden eine feste Definition, die nach der Bereitstellung nicht mehr geändert werden kann. Das garantiert, dass die Metadaten immer existieren, auch wenn die Regelbibliothek auf mehreren Servern zum Einsatz kommt. Zu beachten ist, dass es weiterhin möglich ist, zusätzliche Metadaten über die Webkonsole anzugeben.

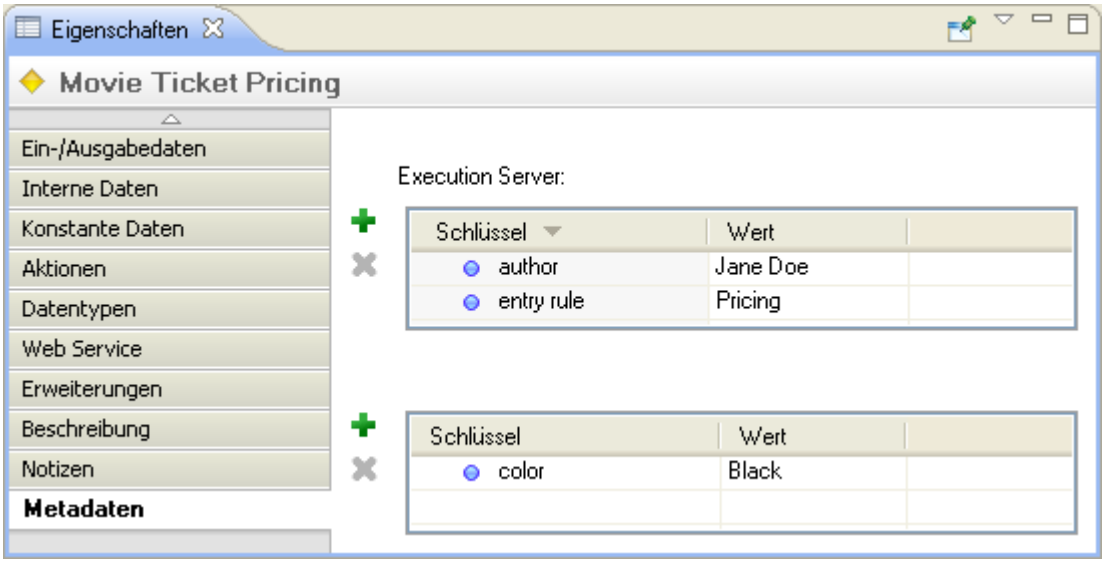
Um Metadaten im Visual Rules Modeler anzugeben führen Sie folgende Schritte aus:

1. Im Projekt Explorer oder Regel Explorer, wählen Sie das Regelmodell aus, welches als Rule Service exportiert werden soll.

2. Wechseln Sie zur Seite Metadaten in der Sicht Eigenschaften.

3. Drücken Sie den Knopf neben der Execution Server Tabelle. Ein Eintrag in der Schlüssel Spalte wird angelegt, mit einem default Namen. Der Schlüssel kann durch einen Doppelklick bearbeitet werden. Mit dem Knopf kann ein Schlüssel und sein Wert gelöscht werden.

4. Mit einem Doppelklick in die Wert Spalte neben dem Schlüssel, kann ein Wert hinzuzufügt oder bearbeitet werden.



Schlüssel und Wert müssen in einem bestimmten Format angegeben werden. Die Länge muss mindestens ein Zeichen, aber maximal 255 Zeichen sein. Leerzeichen sind innerhalb des Elements erlaubt, jedoch nicht am Anfang oder am Ende. Zu beachten ist, dass nur Schlüssel und Werte in der Execution Server Tabelle für die Bereitstellung verwendet werden.

6.2.1. Zuordnung von Metadaten zu Rule Services

Der Execution Server ist in der Lage anhand von Metadaten eine WSDL zu liefern oder eine generische Anfrage zu einem Rule Service zu leiten. Dies wird dadurch bewerkstelligt, dass Metadaten einem bestimmten Rule Service zugeordnet werden. Die Komponente, die dafür verantwortlich ist, heißt `Metadata Mapper`.

Abhängig davon, welche Werte in den Metadaten definiert sind und was in der Anfrage angegeben wurde, kann es dazu kommen, dass es keine eindeutige Zuordnung gibt. Je nachdem, welches Protokoll verwendet wird, führt das in so einem Fall zu einen SOAP oder HTTP Fehler.

6.2.1.1. Standard-Metadata Mapper

Die Standardimplementierung des `Metadata Mapper` benutzt `ruleModel`, `rulePath` und optional `version` für die Weiterleitung einer Anfrage. Damit kann eine generische Anfrage anstelle einer spezifischen verwendet werden, wie das in Abschnitt 4.1.3, „Generische Rule Service Anfragen“ beschrieben wird. Abgesehen davon ist es möglich `effectiveDate` in Kombination mit `ruleModel` und `rulePath` zur Weiterleitung einer Anfrage zu benutzen. In diesem Fall soll ein Rule Service ausgeführt werden, der zu einem bestimmten Datum gültig ist.

Der `rulePath` hat das gleiche Format, wie es in der Generischen Regel Ausführung API verwendet wird, die im Java Integration Handbuch beschrieben ist.

Der Standard-`Metadata Mapper` benutzt die Rule Service Einstellungen `gültig von` (`validFrom`) und `gültig bis` (`validTo`). Ein Rule Service ist gültig, falls `effectiveDate` zwischen den Grenzen von `gültig von` und `gültig bis` (inklusive) befindet. Ist eine der Grenzen nicht gesetzt, wird dies als "immer" interpretiert. Sind beide Grenzen nicht gesetzt, dann ist ein Rule Service immer gültig.

Der Hauptanwendungsfall sind Rule Services in unterschiedlichen Versionen, die zu unterschiedlichen Zeiten gültig sind. Gibt es zum Beispiel ein `Movie Ticket Pricing` in Version 1.0, das ein `gültig bis` auf 2008-12-31 gesetzt hat und es gibt ein weiteres in Version 2.0, dessen `gültig von` auf 2009-01-01 gesetzt ist, dann wird eine Anfrage mit einem `effectiveDate` auf einen Wert vor dem 2009-01-01 die Version 1.0 ausführen und im anderen Fall die Version 2.0.

Aus Gründen der Abwärtskompatibilität werden weiterhin zwei spezielle Metadaten, nämlich `effectiveFrom` und `effectiveTo` unterstützt. Jeder Wert repräsentiert ein Datum und folgt dem Format `yyyy-mm-dd`. Zu beachten ist, dass diese Metadaten nur verwendet werden, wenn weder `gültig von` noch `gültig bis` gesetzt sind.

6.2.1.2. Eine WSDL durch Angabe von Metadata abholen

Der Execution Server erlaubt es, eine WSDL für einen Rule Service durch die Angabe von Metadaten abzuholen. Die dafür benutzte URL hat die Form `http://<server>:<port>/<context-name>/services/wsdl` und Metadaten Elemente werden als Parameter angefügt. Die akzeptierten Werte hängen von der Implementierung des `Metadata Mapper` ab.

Im folgenden Beispiel wird eine Standardinstallation des Execution Servers angenommen, die unter der URL `http://localhost:8080/executionserver-5.x.y/` verfügbar ist. Wegen der Lesbarkeit wurden Zeilenumbrüche eingefügt. Normalerweise würde die URL in einer Zeile eingegeben. Die Platzhalter x und y entsprechen dabei der Versionsnummer aus dem Dateinamen der WAR Datei des Execution Servers.

```
http://localhost:8080/executionserver-5.x.y/services/wsdl
?ruleModel=Movie%20Ticket%20Pricing
&rulePath=/Movie%20Ticket%20Pricing/Pricing
&effectiveDate=2008-01-01
```

In der URL im Beispiel wird eine WSDL für das Regelmodell `Movie Ticket Pricing` mit der Regel `Pricing` angefragt, das gültig ist am `2008-01-01`. Zu beachten ist, dass Sonderzeichen maskiert werden müssen, wenn sie in einer URL verwendet werden.

Beispiel 6.1. Beispiel einer URL mit dem Standard Metadata Mapper

6.2.1.3. Benutzerdefinierter Metadata Mapper

Die Anpassung des `Metadata Mapper` ist ein fortgeschrittenes Thema, da dies die Anpassung der Execution Server Installation, die Erweiterung einer abstrakten Java Klasse und die Änderung einer XML Schema Datei erfordert. Der Vorgang wird anhand eines Beispiels dargestellt.

Der angepasste `Metadata Mapper` wird mit dem Metadatum `rating` (was die Kurzform für "Rating Verfahren" ist), der `version` und dem Metadatum `rulePath` (welches die Regel spezifiziert, die das Rating Verfahren ausführt) arbeiten. Letzteres wird aus technischen Gründen benötigt, weil dies erlaubt eine generische Anfrage zu formulieren ohne das zu verwendende Regelmodell oder Regel angeben zu müssen. Für diesen Fall ist es also erforderlich, dass `rulePath` auf jedem Rule Service definiert wird, der auch `rating` unterstützt.

Das einzige notwendige Element in einer generischen Anfrage ist `rating`. Weil dies unter Umständen nicht ausreicht, um zu entscheiden, welcher Rule Service auszuführen ist, wird im Beispiel auch die Angabe `version` wie auch andere Metadaten verwendet.

Im ersten Schritt muss die `meta.xsd` geändert werden, um valide generische Anfragen zu erlauben. Die Schlüssel und Typen der Metadaten gehören innerhalb des `target` Elements. Der untere Ausschnitt zeigt, wie das aussehen könnte.

```
...
<xsd:complexType name="Target">
  <xsd:sequence>
   <xsd:element name="rating" type="xsd:string" />
   <!-- Jeglicher Wert erlaubt -->
   <xsd:sequence>
     <xsd:any />
   </xsd:sequence>
</xsd:sequence>
</xsd:complexType>
...
```

Als nächstes wird eine Java Klasse `RatingMetaDataMapper` erzeugt, welche die abstrakte Klasse `de.visualrules.execution.core.spi.metadata.AbstractMetaDataMapper` erweitert. Da das Object später per Reflection geladen wird, muss ein default Konstruktur aufrufbar sein. Die abstrakte Klasse fordert die Implementierung von zwei Methoden, nämlich `mapRuleModelArtifact(..)` und `mapRuleInvocationTarget(..)`. Die erste Methode wird aufgerufen, sobald eine WSDL angefragt wird und die zweite, wenn eine generische Anfrage zu einem Rule Service weitergeleitet werden soll.

Die `mapRuleModelArtifact(..)` Methode bekommt zwei Argumente übergeben: die Metadaten als `java.util.Properties` und eine Instanz des `IArtifactStorageReadAccess`. Als erstes wird

geprüft, ob ein Wert für `rating` angeben wurde, ansonsten wird eine Ausnahme geworfen. Danach wird mittels der Metadaten ein `IServiceView` gesucht. Sofern einer gefunden wurde, wird dieser zu einem `RuleModelArtifact` konvertiert und zurückgegeben.

```
public RuleModelArtifact mapRuleModelArtifact(Properties metaData,
      IArtifactStorageReadAccess artifactStorage)
      throws AmbiguousRuleModelArtifactException
{
  String rating = metaData.getProperty("rating");
  if (rating == null)
  {
    throw new IllegalArgumentException("A rating must be set.");
  }

  IServiceView serviceView = findByMetaValues(metaData, artifactStorage);
  return serviceView != null ? new RuleModelArtifact(serviceView) : null;
}
```

Die `findByMetaValues(..)` Methode benutzt die `IArtifactStorageReadAccess#listServicesWithMetaValue(..)` Methode um Rule Services zu finden, in deren Metadaten es einen passenden Wert für den Schlüssel `rating` gibt. Dies kann bei einer großen Anzahl von Services der Fall sein. Zusätzlich werden solche aussortiert, deren Metadaten nicht zu denen passen, die in der Anfrage übergeben wurden. Danach bleibt im günstigsten Fall genau ein `IServiceView` übrig, der zurückgegeben werden kann. Es gibt jedoch auch die Möglichkeit, dass es mehrere Services oder gar keine gibt. In letzterem Fall wird `null` zurückgeben, was impliziert, dass nichts gefunden wurde. Für mehrere Ergebnisse wird eine Ausnahme geworfen, die Informationen zu den gefundenen Services und ihren Metadaten enthält. Für dieses Beispiel macht dies durchaus Sinn. Die Anwendungslogik benutzt die Metadaten der Anfrage um einen passenden Rule Service zu finden. Um genauer zu werden, reicht die Angabe weiterer Werte in der Anfrage. Je spezifischer das wird, desto mehr Services können ausgefiltert werden, bis nur noch einer übrig ist.

Die Art und Weise, wie mit mehreren gefundenen Rule Services umgegangen wird, kann natürlich anders gehandhabt werden, als im Beispiel. Es sollte nur sichergestellt werden, dass es das gleiche Verhalten ist, wie bei der Implementierung der `mapRuleInvocationTarget(..)` Methode.

```
private static IServiceView findByMetaValues(Properties metaData,
      IArtifactStorageReadAccess artifactStorage)
      throws AmbiguousRuleModelArtifactException
{
  String ratingValue = (String) metaData.get("rating");
  if (ratingValue != null)
  {
    List services = filter(artifactStorage.listServicesWithMetaValue("rating", ratingValue),
                           metaData);
    final int size = services.size();
    if (size == 1)
    {
        return (IServiceView) services.get(0);
    }
    else if (size > 1)
    {
        IServiceView[] serviceViews = (IServiceView[]) services.toArray(
                                                       new IServiceView[size]);
        throw new AmbiguousRuleModelArtifactException(
               "There are multiple rule models that fit the requested meta values.",
                serviceViews);
    }
  }
  return null;
}
```

Die `filter(..)` Methode liefert nur die Services, deren Metadaten zur Anfrage passen, was mit der Methode `matches(..)` geprüft wird.

```
private static List filter(List services, Properties metaData)
{
   List retVal = new ArrayList(services.size());
   for (Iterator it = services.iterator(); it.hasNext();)
   {
       IServiceView serviceView = (IServiceView) it.next();
       Map metaValues = serviceView.getMetaValues();
       if (metaValues != null && matches(metaData, serviceView))
       {
           retVal.add(serviceView);
       }
   }
   return retVal;
}
```

Die `matches(..)` Methode vergleicht jedes Schlüssel-Wert Paar aus der Anfrage mit den von jedem `IServiceView`. Die Schlüssel in den Metadaten der Anfrage müssen eine Teilmenge der auf dem Service definierten bilden. Zudem müssen die Werte übereinstimmen. Die einzige Ausnahme ist die `version`, da diese kein Metadatum ist.

```
private static boolean matches(Properties metaData, IServiceView serviceView)
{
    Map metaValues = serviceView.getMetaValues();
    for (Iterator it = metaData.entrySet().iterator(); it.hasNext();)
    {
       Map.Entry me = (Map.Entry) it.next();
       String key = (String) me.getKey();
       String value = (String) me.getValue();

       if ("version".equals(key))
       {
          if (value != null && !value.equals(serviceView.getVersion()))
          {
             return false;
          }
       }
       else if (!metaValues.containsKey(key))
       {
          return false;
       }
       else if (!matchesValue(value, (IMetaValue) metaValues.get(key)))
       {
          return false;
       }
    }
    return true;
}
```

Die `matchesValue(..)` Methode vergleicht nur die Werte unter Berücksichtigung von `null`.

```
private static boolean matchesValue(String value, IMetaValue metaValue)
{
   if (value == null)
   {
      return metaValue.getValue() == null;
   }
   else
   {
      return value.equals(metaValue.getValue());
   }
}
```

Damit ist der notwendige Code geschrieben, um eine WSDL mittels Metadaten abzuholen. Zum Beispiel ist es jetzt möglich, eine WSDL für einen Rule Service anzufragen, der ein Rating Verfahren `PD` benutzt und von `John Doe` erstellt wurde (die Platzhalter x und y entsprechen der Versionsnummer aus dem Dateinamen der WAR Datei des Execution Servers).

```
http://localhost:8080/executionserver-4.x.y/services/wsdl?rating=PD&author=John Doe
```

Der letzte Schritt für das Beispiel ist die Implementierung der Methode `mapRuleInvocationTarget(..)`. Diese wird aufgerufen, wenn eine generische Anfrage an den Execution Server geschickt wird. Weil hierbei etwas ausgeführt wird, spezifiziert der Rückgabewert auch die Regel die ausgeführt werden soll. Im Beispiel wird hierfür das `rulePath` Metadatum verwendet. Die grundlegende Idee ist, dass jeder Rule Service "weiß", was seine Einstiegsregel ist. Das erlaubt die Benutzung von Metadaten für die Weiterleitung einer Anfrage, ohne dass dabei ein Regelmodell oder dessen Regel angeben werden muss.

Der folgende Ausschnitt zeigt die Implementierung der Methode `mapRuleInvocationTarget(..)`. Auch hier wird zuerst die Existenz eines Werts für `rating` geprüft. Danach wird ein `IServiceView` für die Metadaten gesucht. Das erfordert genau die gleiche Zuordnung, wie sie in `mapRuleModelArtifact(..)` verwendet wird, weshalb die vorher geschriebene `findByMetaValues(..)` Methode wiederverwendet wird. Danach wird das Metadatum `rulePath` ausgelesen, ein `RuleInvocationTarget` erzeugt und zurückgegeben.

```
public RuleInvocationTarget mapRuleInvocationTarget(Properties metaData,
      IArtifactStorageReadAccess artifactStorage)
      throws AmbiguousRuleModelArtifactException
{
   String rating = metaData.getProperty("rating");
   if (rating == null)
   {
      throw new IllegalArgumentException("No rating provided");
   }
   IServiceView serviceView = findByMetaValues(metaData, artifactStorage);
   if (serviceView != null)
   {
      Map metaValues = serviceView.getMetaValues();
      if (metaValues != null)
      {
         IMetaValue metaValue = (IMetaValue) metaValues.get("rulePath");
         if (metaValue != null)
         {
            String rulePath = metaValue.getValue();
            RuleInvocationTarget target = new RuleInvocationTarget(serviceView, rulePath);
            return target;
         }
      }
      throw new RuntimeException("No rulePath provided in service: " + serviceView);
   }
   return null;
}
```

Dieses Beispiel beachtet nicht, ob ein Rule Service aktiv ist oder nicht. Damit lassen sich somit auch eigentlich inaktive Rule Services aufrufen. Das kann durch eine Anpassung der `filter(..)` Methode geändert werden, in dem dort inaktive Services ausgefiltert werden.

Als nächstes muss der Execution Server konfiguriert werden, um das Beispiel `RatingMetaDataMapper` zu benutzen. Das wird erreicht durch die Einstellung `visualrules.executionserver.metadata.custom.mapper` wie es in Abschnitt 1.5, „Konfiguration des Execution Server“ beschrieben ist. Der Wert ist der vollqualifizierte Klassenname von `RatingMetaDataMapper`. Mit dieser Konfiguration wird der Execution Server die Klasse beim Hochfahren laden. Es ist daher notwendig, dass sich die Klasse auf dem Klassenpfad befindet. Dies hängt vom eingesetzten Application Server ab. Beispielsweise kann dies im Tomcat erreicht werden, in dem die Klasse in ein JAR gepackt wird und in das Verzeichnis `WEB-INF/lib` der ausgepackten Webapplikation gelegt wird.